UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

TRABAJO FIN DE GRADO

Desarrollo de la versión 2.0 de la aplicación Web de encuestas Yask.it

Estudiante

Bartosz Andrzej Zawada

Tutora

Susana Holgado

Junio 2013

Resumen

Este Trabajo Fin de Grado trata del desarrollo de la aplicación web *Yask.it*.

Yask.it simplifica el proceso de realización de **encuestas dentro de aplicaciones móviles**. Ejerce tanto como componente de realización de encuestas, como sistema de análisis de las respuestas enviadas por los usuarios.

En un principio el proyecto no se centraba sólo en los dispositivos móviles; Su funcionalidad inicial iba dirigida más a tiendas y establecimientos físicos mediante el uso de códigos QR, aunque no resultaba muy difícil usar el sistema para aplicaciones móviles. Con el tiempo se vio que el flujo de datos de aplicaciones móviles era muy superior al de tiendas y se decidió hacer énfasis en la sección de mercado que más éxito tiene.

En cuanto al sistema. Éste permite la creación de encuestas con cualquier número de preguntas, aunque se recomienda no poner más de 6 preguntas.

A la hora de responder, los usuarios de las aplicaciones móviles responden a cada pregunta valorándola con 1 a 5 estrellas (o ninguna si no quieren responder a alguna en particular). Además hay un campo de comentario libre que les permite escribir cualquier cosa que tengan en mente. La razón de este diseño es mantener el **sistema simple y cómodo de usar para evitar espantar a los usuarios**.

Actualmente para responder a las encuestas sólo existe una interfaz web *HTML* que debe ser cargada en un *WebView*. Está planeada la implementación de componentes nativas para las principales plataformas móviles: *Android* y *iOS*. Aunque eso no entra en el ámbito de este TFG.

En particular, casi todo el desarrollo realizado durante este TFG se centra en la página web de *Yask.it*. Se ha agregado importantes funcionalidades de administración y rediseñado la interfaz para atraer clientes. También se han realizado bastantes labores de administración de sistemas informáticos (Servidores dedicados, *Servicios en la Nube*).

Palabras Claves

Servicio Web, Encuestas, Móviles, In-App, Satisfacción de Usuarios, Administración de Sistemas, Análisis de Datos, Computación en la Nube

Summary

This Final Project is about the development of the *Yask.it* web application.

Yask.it simplifies the process of conducting **surveys within mobile apps**. It serves both as component to reply surveys and as user response analyzing system.

Initially, the project wasn't only focused on mobile devices; Its initial functionality was aimed at shops and physical establishments through the use of QR codes, although it wasn't hard to use the system for mobile apps. Eventually we observed that the data flow from mobile apps was well above of that from shops, so we decided to focus on the most successful market.

As for the system, It allows the creation of surveys with any amount of questions, but we recommend to put no more than 6 questions.

When responding, the mobile app users reply to each question by rating it from 1 to 5 stars (or none if they don't want to answer to a question). There is also an open comment field where users can write whatever crosses their minds. The reasoning for this design is to keep the system **simple and comfortable to avoid frightening off the users**.

Currently, for answering surveys, there is only one interface made in *HTML* that has to be loaded into a *WebView*. It's planned to implement native components for the main mobile platforms: *Android* & *iOS*. However this task is not within the scope of this Final Project

In particular, most of the development made during this Final Project focused on Yask.it website. Important administration features were added. A great deal of interface was remade to attract clients. Also there were lots of system administration work done (Dedicated servers, Cloud Services)

Keywords

Web service, Surveys, Mobile, In-App, User Satisfaction, System Administration, Data Analysis, Cloud Computing

Índice

Índice de figuras

Glosario

A/B Testing Metodología de prueba usada en el desarrollo Web. Se basa en disponer de 2 variantes de una interfaz generalmente con diferencias muy ligeras, y ofrecer a la mitad de los visitantes una variante y a la otra mitad la otra variante. Más tarde se decide que variante es mejor revisando en cual de ellas se ha cumplido más veces un objetivo (que el usuario haga click en un *banner*, que compre algo, que se registre, etc)

Active Record Patrón de diseño que se puede encontrar en software que accede a bases de datos. Se basa en que una instancia de un objeto está enlazada a una fila de una tabla en la base de datos. Cuando se crea un objeto, se añade una nueva fila al guardar. Cuando un objeto se modifica, su fila también lo hace.

Coding by convention Patrón de diseño que busca disminuir la cantidad de decisiones que un desarrollador debe tomar, simplificando todo el proceso sin necesariamente perder ninguna flexibilidad.

CoffeeScript Precompilador de Javascript. Basado en identación. Añade gran cantidad de *syntactic sugar*. Previene muchos fallos de codificación en Javascript.

Compass *Framework CSS* desarrollado en *SASS*. Añade muchas funcionalidades bastante interesantes, que simplifican seriamente el desarrollo de CSS que funcione para todos los navegadores. Por ejemplo, permite escribir una sola regla que se compilará en todas las reglas prefijadas para cada navegador.

DRY *(Don't Repeat Yourself)* Patrón de diseño que promueve la eliminación de la duplicación en el código fuente o los datos.

EC2 (*Elastic Compute Cloud*) Servicio web suministrado por Amazon semejante a los VPS. Se basa en el alquiler de servidores virtuales, cuya particularidad es la capacidad de ejecutar una instancia de un servidor casi instantáneamente (de ahí el nombre *Elastic*). Es un sistema muy bueno en caso de que se tenga un tráfico muy inestable, ya que permite encender instancias de servidores cuando el tráfico sea alto, y apagarlos cuando no sean necesarios. Por otra parte sufre de un rendimiento muy bajo, y un coste muy elevado.

Framework Plataforma software usada para desarrollar aplicaciones. Suelen incluir programas de soporte, compiladores, librerías, herramientas y APIs que ayudan en el proceso de desarrollo.

GUID (*Globally Unique Identifier*) Se trata de un identificador generado por un sistema. Debe ser suficientemente aleatorio para que jamás ocurra una colisión con otro GUID.

HAML *(HTML abstraction markup language)* Es un preprocesador de HTML, su principio de funcionamiento es "el marcado debería ser elegante". Utiliza la identación del código para mostrar la estructura y está intrínsecamente relacionado con Ruby, permitiendo ejecutar código. Frente a HTML tiene la ventaja de ser extremadamente más sencillo de leer, más rápido de escribir y ocupa mucho menos espacio.

iframe Elemento *HTML* que sirve para cargar una página web dentro de otra. Suele tener ciertas restricciones de interoperabilidad en los navegadores para mantener la seguridad del usuario.

jQuery Librería de Javascript diseñada para simplificar la escritura de código en el lado del cliente. Además abstrae algunas funcionalidades que diferían entre distintos navegadores, permitiendo escribir código multi navegador de forma rápida y fácil.

MVC *(Model View Controller)* Patrón de diseño que especifica que se debe separar la lógica del sistema, de la interfaz de usuario y del módulo encargado de gestionar eventos.

Nginx	Servidor web basado en eventos. Hace la competencia a Apache. Últimamente está ganando debido a que permite mayor *thoughtput* y consume menos memoria.
Phusion Passenger	Servidor de aplicaciones que se comunica con el servidor web (*Apache*, *Nginx*) y administra las instancias de *Ruby* o *Python* para responder a las peticiones web.
Profiling	Análisis de rendimiento de software.
Ruby	Lenguaje de programación dinámico, orientado a objetos, de propósito general. Basado en *Perl*, *Smalltalk*, *Eiffel* y *Lisp*. Soporta múltiples paradigmas de programación: Funcional, Orientado a objetos e Imperativo.
Ruby on Rails	*Framework* de desarrollo de aplicaciones web escrito en Ruby. Permite crear de forma muy sencilla aplicaciones que enruten las peticiones, se comuniquen con bases de datos, generen el html de las páginas web. Enfatiza ciertos patrones de diseño, entre ellos: Active Record, *MVC, DRY* y *Coding by convention*.
SASS	*(Syntactically Awesome Stylesheets)* Es un preprocesador de CSS, permite utilizar funcionalidades que no existen en CSS (variables, bucles, listas, condicionales, reglas anidadas, herencia de seleccionadores, mixins). Tiene dos sintaxis diferentes, SCSS, y SASS. SCSS es un superconjunto de CSS, que simplemente añade funcionalidades SASS es una sintaxis inspirada en la sintaxis de HAML, usando identación y claridad visual.
SEO	(*Search Engine Optimization*) Proceso para que los buscadores web muestren la página antes que otras.

(Spam) Bot Software automatizado cuyo objetivo es rellenar cualquier campo de texto que encuentre un formulario web con basura y enviarlo. Generalmente para publicitar alguna web ajena, o intentar destruir la puntuación SEO de la web atacada.

Syntactic Sugar Sintaxis dentro de un lenguaje para hacerlo más claro, conciso o elegante.

UserAgent Identificación que suministra cada aplicación que accede a la *World Wide Web* a las webs visitadas.

VPS (*Virtual Private Server*) Se trata de una máquina virtual ejecutándose en un servidor. Suelen ser ofrecidos por Servicios de *hosting* en internet. La diferencia frente al *Web hosting* común, es que el cliente es el responsable de administrar el sistema. Esto complica las cosas para gente sin experiencia, pero es bueno para gente experimentada porque permite la instalación de cualquier software que puede no ser ofrecido en un web *hosting* común.

Webkit Motor de renderizado de páginas web. Usado por los navegadores *Google Chrome* (hasta la versión 27) y *Apple Safari* entre otros. Además los navegadores nativos de los dispositivos *Android* y los dispositivos *iOS* usan este motor.

Introducción

Este TFG trata sobre el avance en el desarrollo de la aplicación web de encuestas Yask.it.

El objetivo del trabajo es realizar un progreso en el proyecto de forma que se acerque mas a ser un producto comercial, que ofrezca una imagen seria y de confianza para que los desarrolladores de aplicaciones móviles se atrevan a probarlo en sus aplicaciones. Esto se consigue tanto mejorando lo que hay actualmente hecho como añadiendo nuevas funcionalidades.

La motivación para este proyecto proviene del interés por desarrollar un software que sea utilizado en el mundo real por gente para propósitos reales. Obtener información sobre lo que piensan tus usuarios de tu aplicación móvil parece un buen ejemplo de tal uso. Además es un proyecto personal de gran escala que en un futuro puede resultar muy útil para demostrar mis capacidades de desarrollador.

La estructura de la memoria es bastante sencilla. Primero se explicará el proyecto base. Sus principios y su funcionamiento. Luego se explicará parte a parte, todo lo que se ha añadido o cambiado. En caso de cambios, se mostrará como era antes y en que se ha convertido.

Proyecto base: Yask.it

Idea y Diseño

Yask.it surgió como idea hace más de un año. Inicialmente sería un sistema informático para que tiendas, restaurantes y otros establecimientos físicos pudieran realizar encuestas de satisfacción a sus clientes de forma rápida y cómoda tanto para los clientes como para los establecimientos.

Este sistema se podría implementar de diversas maneras:

- En forma de aplicación móvil que los establecimientos instalarían en hardware dedicado a tal fin, como por ejemplo *iPads* u otras *Tablet*. El empleado del establecimiento pasaría el dispositivo a los clientes para responder a la encuesta. Luego las respuestas podrían o bien almacenarse en el dispositivo para ser revisadas por el establecimiento (lo que resulta poco práctico) o podrían ser enviadas a un servidor en internet que las acumulara. Este enfoque tiene una desventaja bastante importante:

    - La compra de los dispositivos podría resultar en un desembolso importante por parte de los establecimientos. Desembolso al que, probablemente, la mayoría de establecimientos no estuviera dispuesta.

- En forma de aplicación móvil que los clientes instalan en su teléfonos y que desde la app se respondiera a las encuestas de los establecimientos. Las respuestas luego serían mandadas a un servidor en internet que permitiese a los establecimientos ver los resultados. Este enfoque soluciona los problemas del enfoque anterior, ya que la mayor parte de la población dispone de un teléfono móvil o *tablet*, que además porta consigo. Pero incluye otros problemas muy grandes:

    - Responder a la encuesta requiere conexión a internet.

    - Los clientes deben instalar una aplicación para responder a las encuestas. Es muy probable que algún cliente quiera responder a la encuesta, pero instalar una app en su móvil le resulte una exigencia demasiado grande y por ello desista.

- En forma de servicio Web que se pueda mostrar en cualquier dispositivo. Este enfoque es la mezcla de ambos anteriores, ya que permite que los establecimientos pasen a los clientes un dispositivo con la encuesta cargada, o que los clientes la abran en sus dispositivos móviles y respondan. Evitando el desembolso para el establecimiento que no esté dispuesto y no obliga a los clientes a instalar nada, ya que casi todos los dispositivos móviles actuales disponen de navegador web. Sin embargo, el requisito de conexión a internet sigue estando presente.

Para *Yask.it*, se ha optado por el tercer enfoque. Un servidor central gestionaría las encuestas de todos los establecimientos.

Sistema de encuestas

Ante todo, el sistema entero debe ser **extremadamente simple y fácil de usar**, tanto para nuestros usuarios (establecimientos) cuyo nivel de conocimiento tecnológico puede no ser muy elevado; como para sus clientes, que es más probable que respondan a las encuestas si éstas no son ni largas, ni complicadas.

Por ello, la página de la encuesta es extremadamente simple, sólo se admite un tipo de pregunta que se responde valorando con 1 a 5 estrellas. Esto es debido a que es muy rápido y sencillo de responder en una pantalla táctil. Había planes para añadir más tipos de preguntas pero no resulta prioritario. Ahora bien, al estar dirigido a encuestas de satisfacción, debe haber algo que permita a los clientes expresarse libremente. Para este fin, todas las encuestas contienen un campo de comentario libre totalmente opcional.

Cada establecimiento tendría una URL propia a la que los clientes se conectarían para responder a la encuesta. Los clientes podrían responder a la encuesta de forma anónima y sin dar la cara, lo que al menos teóricamente, permite que respondan con mayor sinceridad.

Luego los establecimientos entrarían a la página web de *Yask.it*, donde podrían acceder a las respuestas de sus usuarios, que además, ofrece estadísticas interesantes sobre los datos recogidos para los establecimientos.

El sistema ofrece otra funcionalidad muy útil también, y es que al funcionar todo en la web permite realizar un cambio instantáneo de un cuestionario a otro. Por tanto si en algún momento el establecimiento decide cambiar su campaña de encuestas y preguntar cosas distintas, podrá realizarlo e instantáneamente sus clientes estarían respondiendo a la encuesta nueva. Y como es de esperar, los datos no se mezclarían.

Cómo hacer llegar las encuestas a los clientes

Como se desea que el sistema sea lo más simple posible, era necesario encontrar una forma de que los clientes pudieran cargar sus encuestas en sus teléfonos cómodamente. Escribir una URL en el teclado de un móvil es una tarea bastante molesta. La solución era el uso de códigos QR. Éstos serían escaneados por los clientes con sus dispositivos móviles, abriendo así la página de las encuestas para responder. Esta propuesta vuelve a traer consigo el problema de que requiere una aplicación móvil que escanee códigos QR. Pero ésta es más genérica así que resulta aceptable.

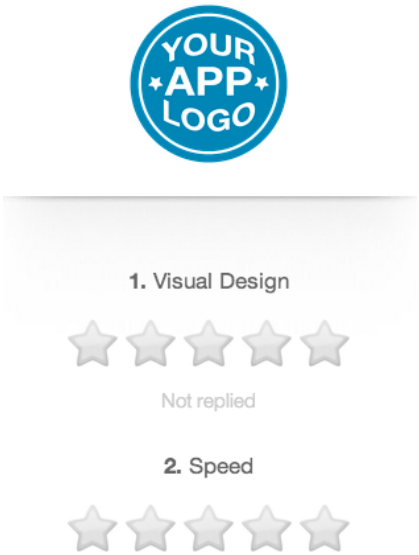

Fig 1. Ejemplo de página de encuesta

Al final lo que se hizo era diseñar unas pegatinas que se esperaba que los establecimientos pegasen en sus vitrinas. Estas pegatinas incluían el *logo* del establecimiento, la URL de la encuesta (para aquellos que no dispongan de la posibilidad de escanear códigos QR o prefieran escribirlos), un código QR bastante grande y una frase que incitara a los clientes a realizar la encuesta.

Fig 2. Ejemplo de pegatinas con códigos QR obtenidas de imprenta profesional

Origen del nombre

Un tema muy interesante es que el sistema pasó bastante tiempo sin tener nombre (no empezó llamándose *Yask.it*). Obtuvo el nombre cuando se buscó un dominio donde alojar el sistema. Tras múltiples intentos de conseguir un dominio corto, fácil de pronunciar y escribir, y que no estuviera ya comprado, se optó por Yask.it. En sí Yask.it proviene de mezclar *Ya + Ask + It*. Representa la intención de obtener respuestas.

Otros usos del sistema

Otro tema muy interesante es que el sistema de encuestas al ser una página web diseñada para ser utilizada en cualquier dispositivo, permitió extender el marco de funcionalidad a Yask.it a otros ámbitos, como encuestas en aplicaciones móviles, encuestas por Email e incluso encuestas en página web. Este detalle es muy importante para entender la evolución del proyecto en el tiempo.

Respuestas repetidas

Como es un sistema de encuestas, hay que mantener control sobre los usuarios finales (encuestados). Nadie quiere que venga un usuario y mande múltiples respuestas para amañar los resultados. La causa de semejante comportamiento no es relevante para el sistema. Sin embargo éste debe poder intentar protegerse. Esta defensa puede resultar muy complicada en la web.

Para poder detectar si alguien está respondiendo múltiples veces, hay que poder identificar a los usuarios. Para ello hay varias soluciones:

- Obligar al usuario a dar alguna información personal. Por ejemplo dirección de correo electrónico, usuario de Twitter o Facebook, o cualquier otro dato que sea identificativo. Este enfoque es imposible para Yask.it, porque requiere trabajo por parte de los usuarios finales, lo que choca directamente con el principio de sencillez y comodidad al que se cierne el sistema.

- Etiquetar al usuario automáticamente. Traducido a un sistema web se basa en usar *cookies* de navegadores. El sistema genera una GUID para cada usuario y la almacenará en una *cookie* en su navegador. Cuando un usuario responda a una encuesta, el sistema guardará la GUID asociada a dicho usuario junto a la respuesta. En caso de que aparezca otra respuesta con la misma GUID se asume que ha sido enviada por el mismo usuario.

- Identificar a un usuario a base de factores terciarios (navegador, *UserAgent*, extensiones del navegador, hardware). Sería necesaria implementar un sistema que basándose en esos datos identificase inequívocamente a un usuario. Tal sistema existe, se llama *Panopticlick*[1] , pero es demasiado exagerado implementar algo semejante en Yask.it.

Por tanto en Yask.it, se ha optado por la segunda fórmula. Ahora bien, las *cookies* tienen un problema y es que (por suerte y por desgracia) se pueden borrar. Si un usuario borrase sus *cookies* el sistema no podría identificarle y le trataría como a un usuario nuevo. Se buscó soluciones para esto, y se encontró *Evercookie*[2]. Se trata de un sistema escrito Javascript cuyo objetivo es generar *cookies* que intencionalmente son complicadas de eliminar. Lo consigue a base de guardar la *cookie* en todos los lugares posibles de un navegador. En caso de que alguna *cookie* falte, será regenerada a partir de las demás. Pero este sistema no funcionó porque los navegadores mas recientes se deshacen de todas las *cookies* a la vez exactamente para impedir el funcionamiento de *evercookie*. Finalmente se dejó el sistema con *cookies* básicas. Identifica a los usuarios que envíen respuestas repetidas e impide la repetición de respuestas básicas. En general el sistema de detección es de mejor esfuerzo.

Es interesante explicar como reacciona el sistema ante respuestas repetidos. En principio sólo interesan las respuestas más recientes de cada usuario. Lo que se hace es si la anterior respuesta fue enviada en los últimos 15 minutos, se asume que no tiene ningún valor y se elimina la anterior, dejando la nueva. Esto protege frente a los navegadores que al recargar una página o pulsar el botón de atrás, reenvían el formulario de la encuesta.

En el otro caso en el que la anterior respuesta fuera enviada hace más de 15 minutos, ésta se “archiva”, y la nueva se sobrepone. Esto se hace para mantener un histórico de cambios. En un futuro se implementará una interfaz que muestre la evolución de las respuestas enviadas por los mismos usuarios. Permitiendo ver si se han corregido los problemas, o si algo ha empeorado con el tiempo. Un usuario que ya hubiera respondido antes sirve muy bien para hacer una comparativa en el tiempo.

Control de SPAM

Yask.it también debe protegerse del spam. El sistema de protección anti spam implementado está bastante relacionado con el de identificación de usuarios. Básicamente, el sistema usa las GUID de los usuarios. Las GUID realmente no se asignan cuando un usuario envía una respuesta. Sino antes, cuando el usuario entra en la página.

Según se carga la página Javascript realiza una petición al servidor. Esta petición lleva integrada la *cookie* que interesa. El servidor responde con la misma GUID, o una GUID nueva si el usuario no tiene ninguna asignada. Luego el cliente lee la respuesta del servidor y guarda la GUID en la *cookie*.

Ahora para bloquear las respuestas que se consideren inválidas, el servidor requiere que todas las respuestas enviadas tengan una *cookie*. Si una respuesta viene sin *cookie*, es que no ha ejecutado el Javascript anterior y se supone que ha sido enviada por sistema automatizado y se rechaza.

Esta protección es bastante básica. Pero sirve ante clientes que no dispongan de Javascript o *cookies*, lo que debería repeler la mayoría de clientes simples. No hay razón para que un *bot* mas avanzado atacase Yask.it, el texto enviado no es públicamente legible.

Internacionalización

Yask.it soporta internacionalización, la página web en sí está escrita tanto en Inglés como en Español. Sin embargo, las encuestas soportan todos los idiomas que sean requeridos. Los clientes escriben sus cuestionarios en su idioma principal, y luego traducen a cualquier otro que crean necesario. El idioma en que se presenten las encuestas se intenta detectar a partir de la configuración del navegador, en caso de que el idioma del navegador no esté entre las traducciones, se usará el idioma por defecto. También es posible forzar el idioma de la página mediante la URL aunque eso es una funcionalidad menos importante.

Desarrollo y despliegue

En cuanto al despliegue de Yask.it. El sistema empezó siendo desplegado en un VPS de bajo coste económico. Debido a que es un servicio con muy baja cantidad de flujo de datos. No hacía falta más. Pero el VPS que estaba contratado empezó a tener problemas de disponibilidad bastante continuados y al final se decidió migrar al servicio de *Elastic Cloud Computing* de Amazon debido a una oferta de un año gratis de una instancia.

El sistema por dentro está escrito en *Ruby on Rails*. Usa algunas tecnologías interesantes como *HAML*, *SASS, Compass*, *CoffeeScript y jQuery*. Estas tecnologías son punteras en el desarrollo web a día de hoy. Además, se utilizaba Apache como servidor web, MySQL como base de datos y *Phusion Passenger* como servidor de aplicaciones.

Cambios realizados durante el TFG

En esta sección de la memoria se describirá los cambios realizados, uno a uno, al sitio web Yask.it durante este TFG.

Cambio de Objetivos

Lo primero que hace falta decir antes de hablar de los cambios en el proyecto durante el TFG. Es explicar que se produjo un cambio de objetivos. Inicialmente, el sistema iba enfocado a realizar encuestas en tiendas y establecimientos físicos. Siendo a la vez lo suficientemente flexible para permitir realizar encuestas en aplicaciones móviles.

Después de un año entero funcionando, el sistema ha acumulado 1780 respuestas. El 85% de las respuestas proviene de 2 aplicaciones móviles, y el 15% restante de múltiples tiendas y talleres.

Los datos se muestran a la derecha, aunque son la cabecera únicamente, se puede ver claramente la abismal diferencia.

Correo electrónico	Respuestas
Aplicación Móvil	779
Aplicación Móvil	732
Tienda	71
Tienda	63
Ejemplo Yask.it	48
Tienda	20
Online	19
Aplicación móvil en desarrollo	11
Aplicación móvil en desarrollo	10
Tienda	9
Tienda	8
Tienda	7

Fig 3. Lista de cuentas con su número de respuestas

Las aplicaciones móviles tienen un flujo de datos muy superior al de las tiendas. Incluso habiendo muchísimas más tiendas (más de la mitad no llegan a aparecen en esa lista, porque tienen un número de respuestas muy bajo)

Teniendo estos datos en cuenta, la acción obvia es redefinir el objetivo del proyecto. Si antes iba enfocado a tiendas y establecimientos físicos, cambiar el enfoque hacia las aplicaciones móviles. Ya que éstas han demostrado que el producto funciona, mientras que las tiendas no.

Este cambio tiene ciertas ventajas adicionales. Antes había que convencer a cada propietario de establecimientos en persona. Aunque solían aceptar probarlo (sobretodo siendo gratis). Se imprimían las pegatinas y muchas veces acababan por no colocarlas, lo que resultaba una pérdida de tiempo y dinero. Sobretodo porque la imprenta resulta cara cuando no imprimes en muy grandes cantidades. En cambio ahora, se debe poder publicitar el sistema mediante campañas online, más fáciles de realizar.

Página Principal

El cambio de enfoque de proyecto se decidió justo antes de reescribir toda la página principal. Por esta razón la nueva página muestra un cambio muy claro tanto en el diseño como en los conceptos que intenta transmitir y su forma de hacerlo.

El anterior diseño era este:

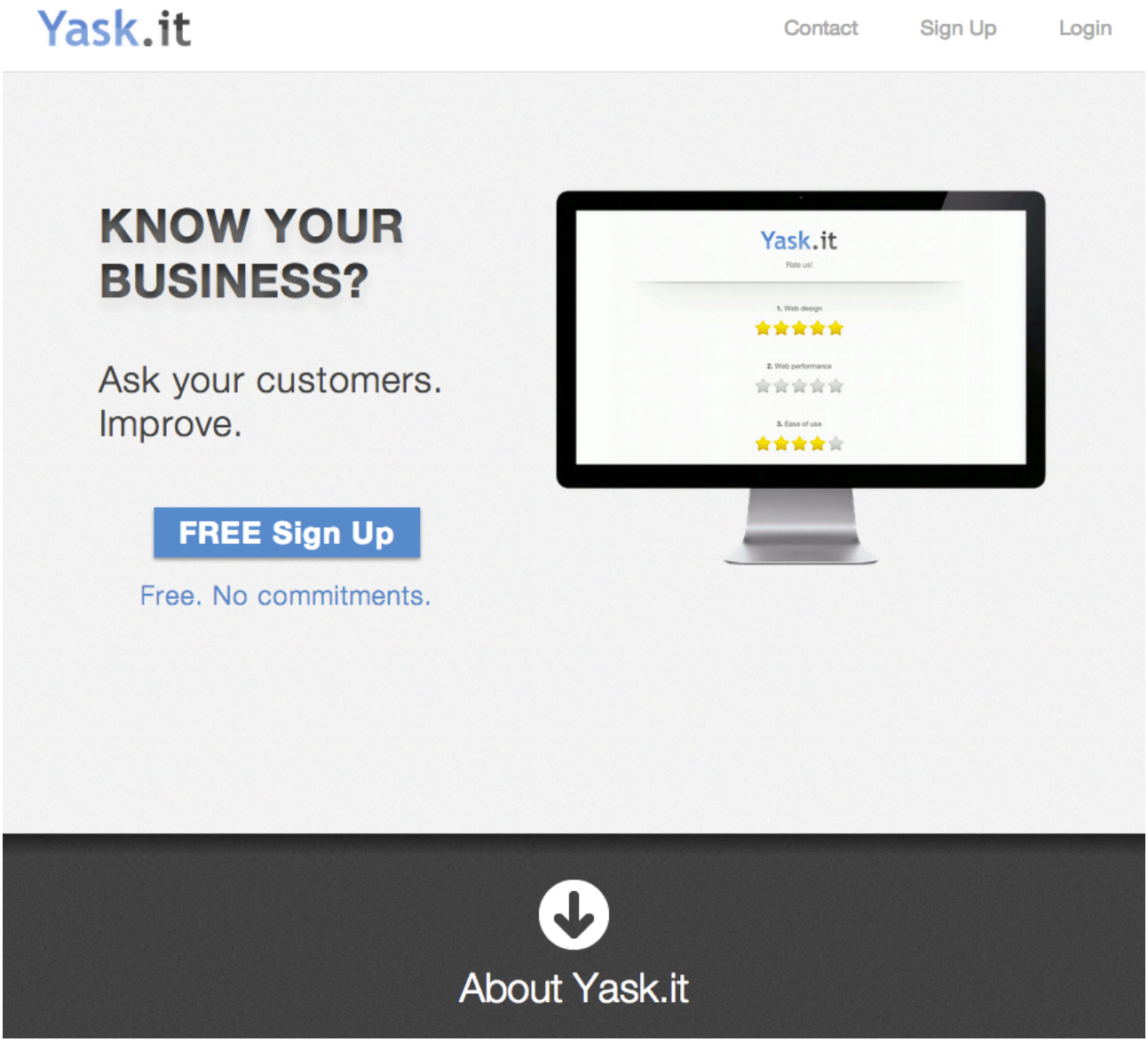

Fig 4. Parte superior del diseño viejo de la página principal

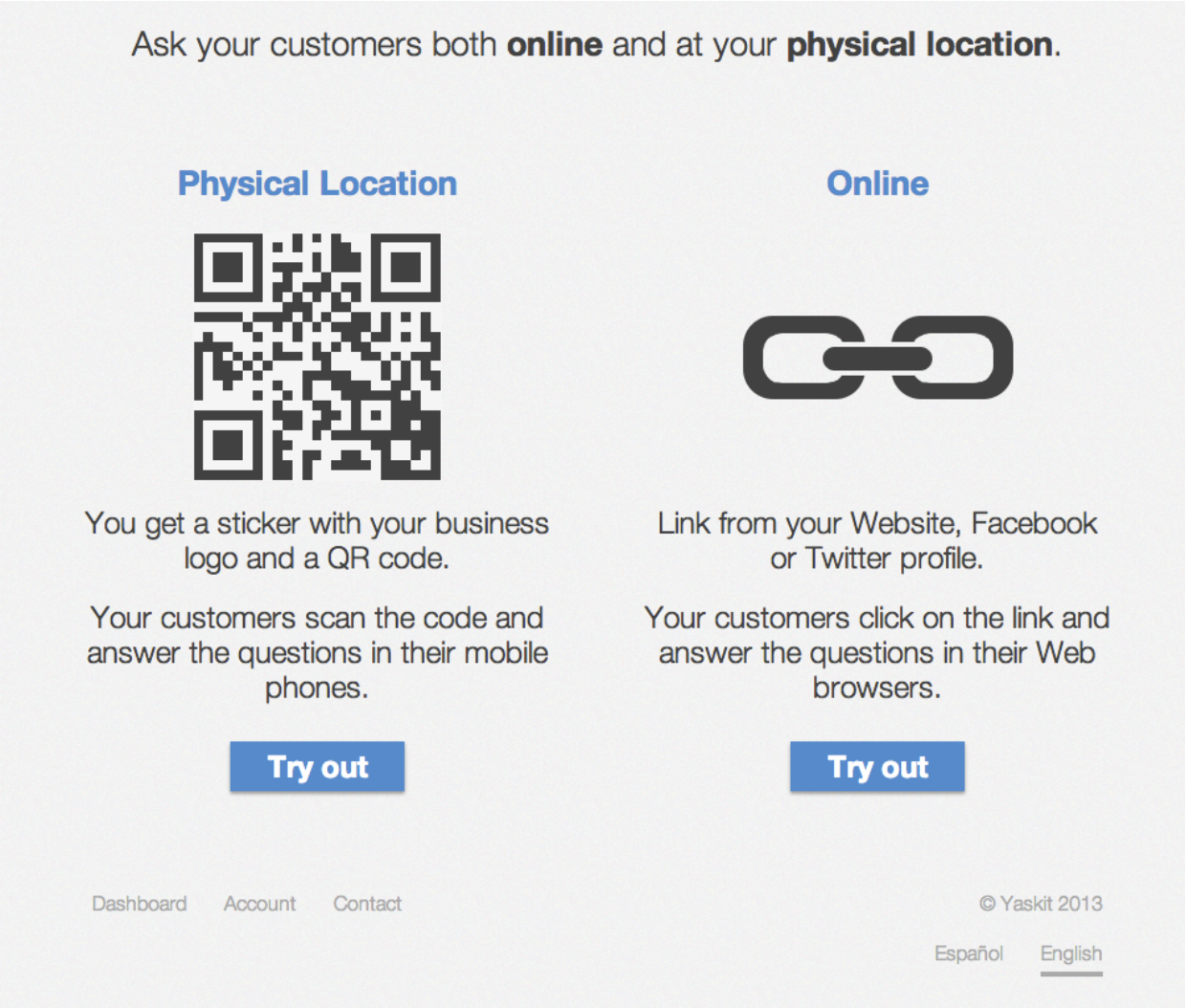

Fig 5. Parte inferior del diseño viejo de la página principal

Los pequeños detalles del diseño que no se aprecian en las imágenes son:

- La imagen del ordenador con la encuesta, cambia cada varios segundos para mostrar una imagen de dos móviles con encuestas abiertas.

- Al hacer click sobre la barra "*About Yask.it*" la pantalla baja para mostrar la parte inferior.

Este diseño lleva así básicamente desde los principios del proyecto. No era muy atractivo así que era hora de que sufriera un cambio importante.

A la hora de crear el nuevo diseño de la página principal se ha seguido los diseños de *startups*:

La página entera se divide en varias secciones, cada una enfocada a una cosa. En la barra superior de navegación se incluyen tres vínculos a distintas secciones importantes de la página.

La primera sección suele ser una imagen del producto, en nuestro caso encuestas en aplicaciones móviles, con un texto muy breve que explica su funcionalidad y una foto bonita de fondo. Es mejor si el fondo tiene alguna relación con el producto, aunque esto no es estrictamente necesario.

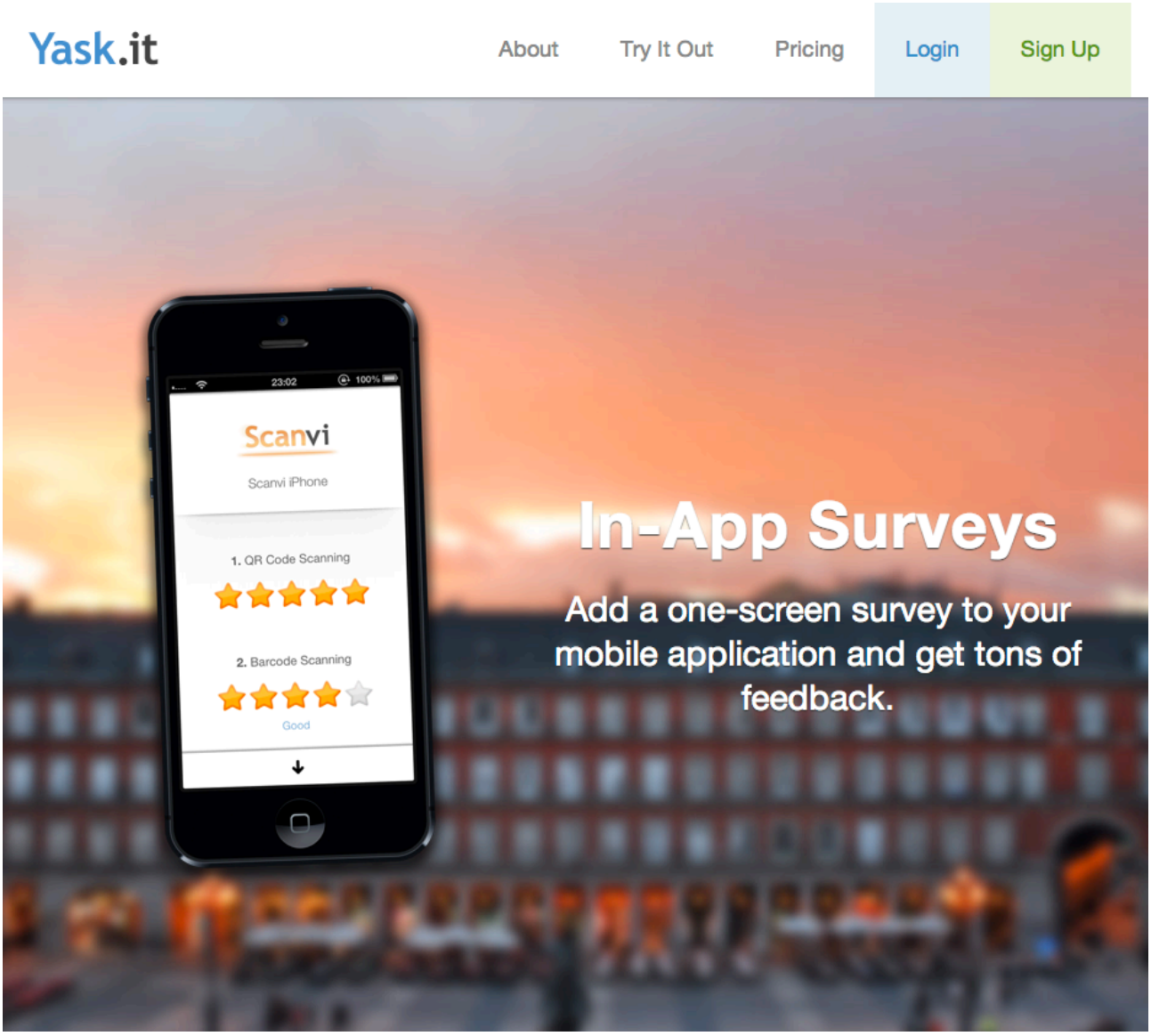

Fig 6. Primera pantalla de la primera sección de la nueva página principal

Esta sección, además, cambia cada varios segundos para mostrar más texto y otra imagen.

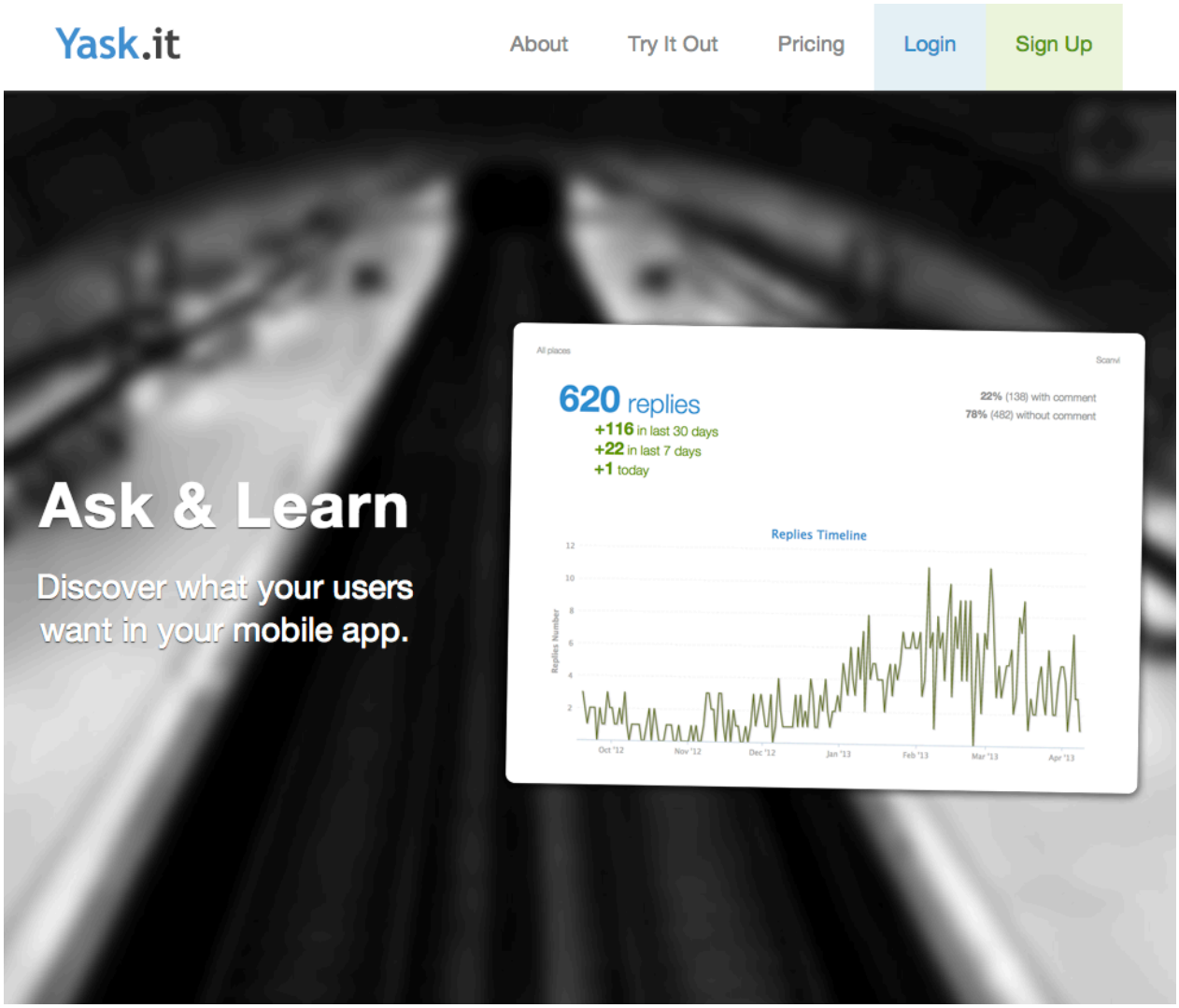

Fig 7. Segunda pantalla de la primera sección de la nueva página principal

Ambas imágenes de fondo son fotos de Madrid con un filtro de difuminado aplicado. Al estar difuminado el fondo, el texto y la imagen principales quedan remarcadas.

Como pequeño detalle, al pasar el ratón sobre el teléfono o la tarjeta, éstos se giran, otorgándole cierta vida a la página web.

Además, en caso de que la pantalla tenga poco ancho (menos de 1100px) el texto y las imágenes se reposicionan uno encima del otro para caber bien.

Después de una primera sección visualmente atractiva, la segunda sección es informativa. Muestra una descripción más profunda del servicio.

Uno de los objetivos más importantes de la página principal de un servicio es vender el producto a los visitantes. Para intentar cumplir este objetivo esta sección se centra en transmitir las siguientes ideas:

- El sistema es muy sencillo de instalar y usar.
- No desempeña la misma labor que otros servicios (Analíticas).
- Usando el sistema obtienes mucha más información sobre tus usuarios.

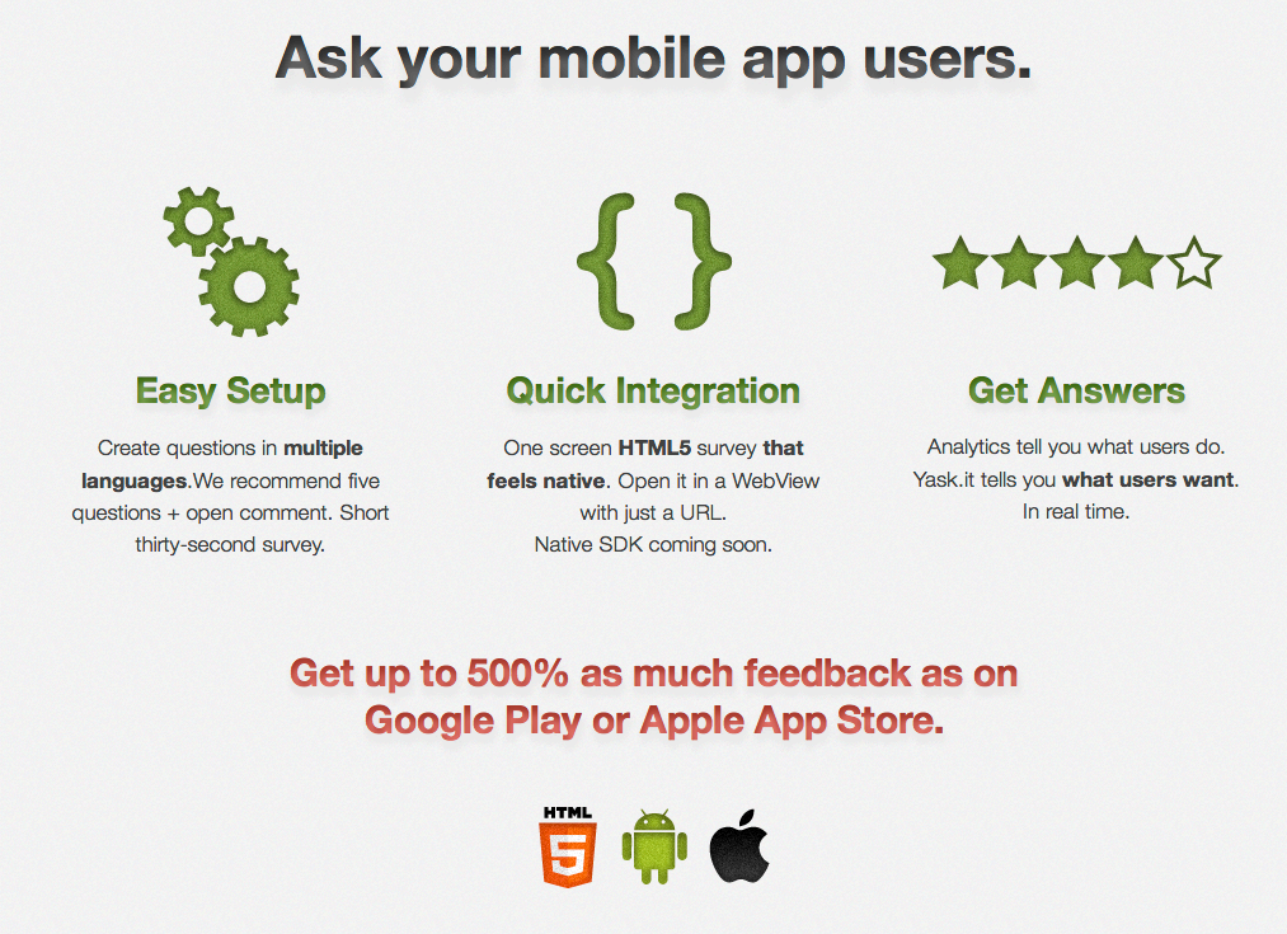

Fig 8. Segunda sección de la nueva página principal

Después de la sección informativa, es importante hacer una demostración del producto.

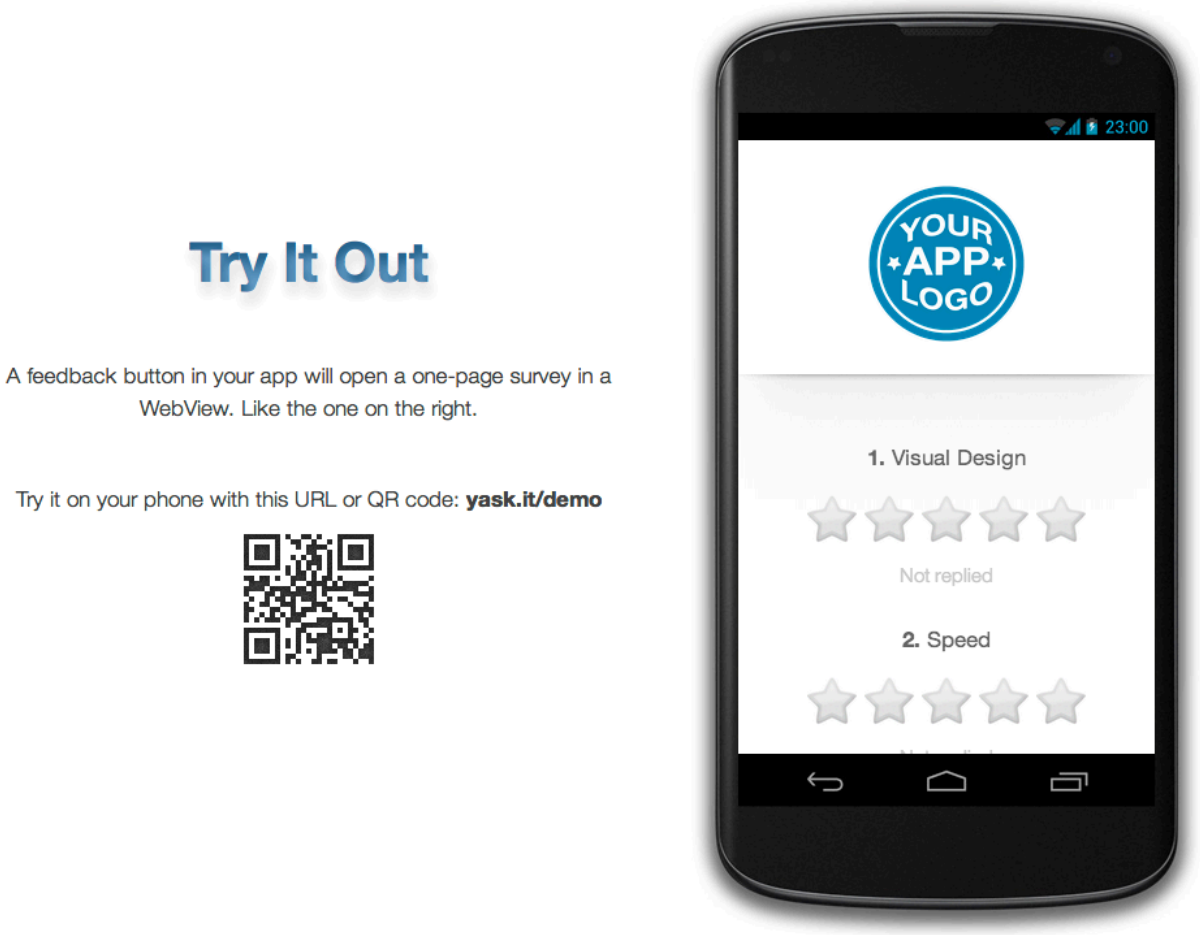

Fig 9. Tercera sección de la nueva página principal

Esta sección ofrece una muestra en vivo de una encuesta. La experiencia cambia dependiendo de si se usa un dispositivo móvil o un ordenador de escritorio.

En un dispositivo móvil lo que se puede ver es un botón que abre la encuesta a pantalla completa porque la idea de mostrar un teléfono móvil virtual dentro de un dispositivo móvil resulta un poco absurda, y además problemática; Los navegadores web en *iOS* no permiten controlar el tamaño de los *iframes*, lo que impide trabajar bien con ellos.

En un ordenador de escritorio, por otra parte, se puede ver exactamente lo mismo que se aprecia en la Fig 9. El contenido del móvil es interactivo.

Funciona como un dispositivo móvil. Al hacer clic es como si pulsaras con el dedo. Así que para desplazarte por la encuesta hay que hacer clic y arrastrar. Es posible responder a la encuesta en este teléfono virtual, en cuyo caso la encuesta se recargará tras unos pocos segundos para que se pueda seguir probando.

La implementación de este "teléfono" virtual resultó ser extremadamente difícil porque los navegadores suelen controlar de forma muy seria cualquier interacción de una página web con otras páginas a través de *iframe*. Era necesario básicamente controlar el contenido del *iframe* desde fuera, lo que incluye meter todo el contenido de la página de la encuesta en un elemento que hiciera scroll, y usar un *plugin* especial para controlar la interacción de hacer clic y arrastrar.

Además, el teléfono virtual puede causar problemas con SSL si no se tiene suficiente cuidado. La página principal siempre requiere SSL, mientras que en la página de la encuesta esto es opcional. En caso de que se cargara la página de encuesta sin SSL, el navegador se negaría totalmente a permitir cualquier acceso desde Javascript al contenido del *iframe* debido a las políticas de mismo dominio. Estas políticas prohiben cualquier interacción entre distintos dominios (el protocolo cuenta) por seguridad.

Tras la sección de prueba, se muestra la sección de precios. Se ha aprovechado para mostrar un listado de funcionalidades actuales y planeadas para que los visitantes sepan que esperar de Yask.it en el futuro.

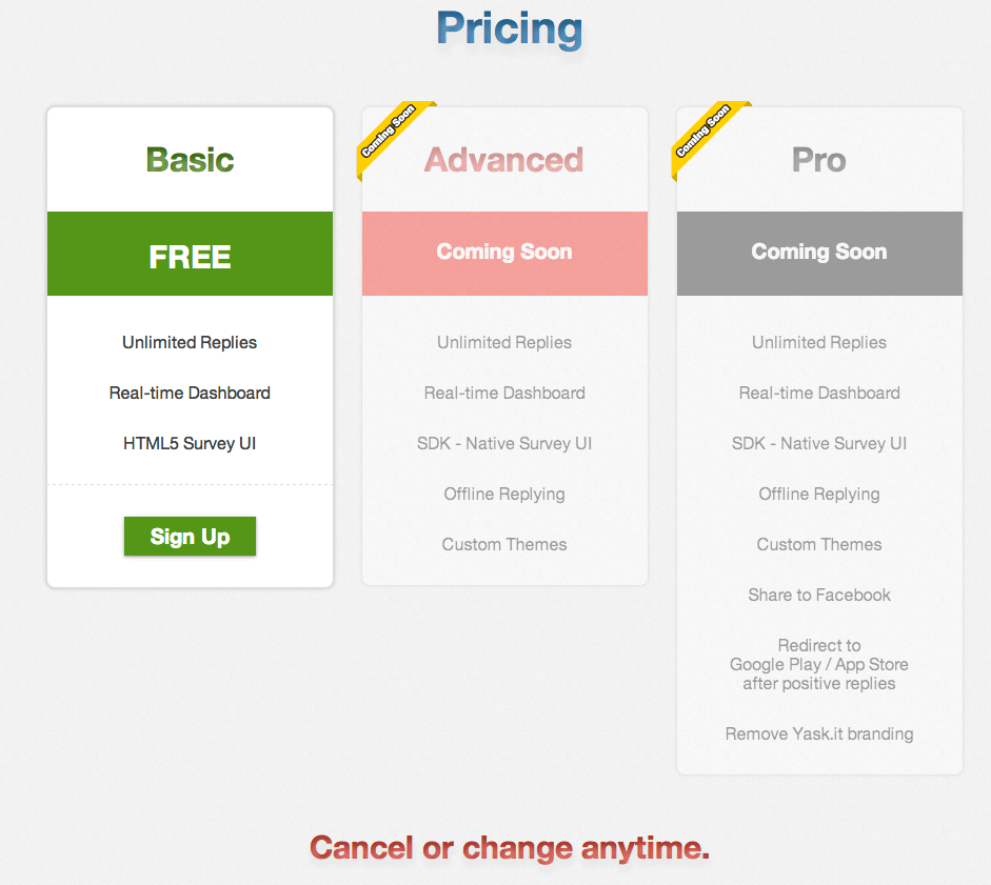

Fig 10. Cuarta sección de la nueva página principal

Como modelo de negocio para Yask.it, se ha pensado en un modelo *Freemium*. Se ofrece una versión gratuita limitada en funcionalidades y luego más opciones de pago. Aunque esto está aún bajo discusión. Frente a *Freemium* está la otra posibilidad. Todo de pago pero ofreciendo una temporada de prueba.

Por un lado, *Freemium* tiene la ventaja de que es gratis. Al ser gratis teóricamente hay más gente que lo probará, lo que es bueno para ganar usuarios iniciales.

Por otro lado, un servicio puramente de pago recauda más dinero. Además es posible que muestre una imagen de producto más sólido, que no vaya a desaparecer en cuestión de varios días, lo que puede resultar interesante.

El precio de las dos versiones de pago aún no está establecido.

Por último, la última sección de la página realiza la llamada a la acción, que en este caso es registrarse en el sistema para empezar a usarlo. Se caracteriza por el enorme botón que desentona con el resto de toda la página para llamar la atención.

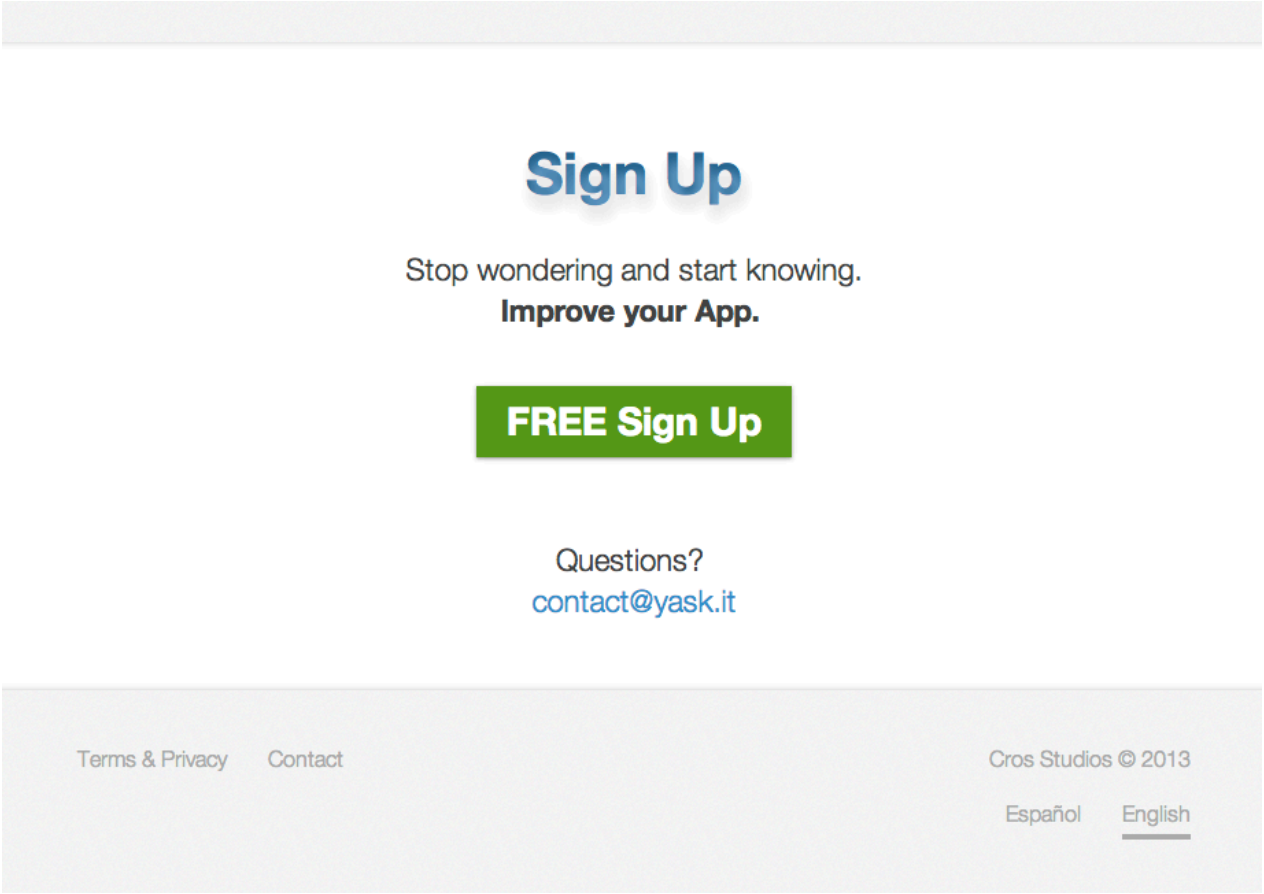

Fig 11. Quinta sección de la nueva página principal

En conclusión, la nueva página principal tiene un diseño mucho mas profesional. Su objetivo es hacer que los visitantes consideren el proyecto como algo serio. Con el anterior diseño era complicado considerar eso. Se espera que el nuevo diseño ofrezca mucho mejores resultados.

Con el cambio de la página principal, dos secciones han desaparecido totalmente.

Primero, la sección de contacto que antes estaba tanto en la barra superior de navegación como en la pie de página. Al hacer click, se cargaba la página de contacto que mostraba algo semejante a esto:

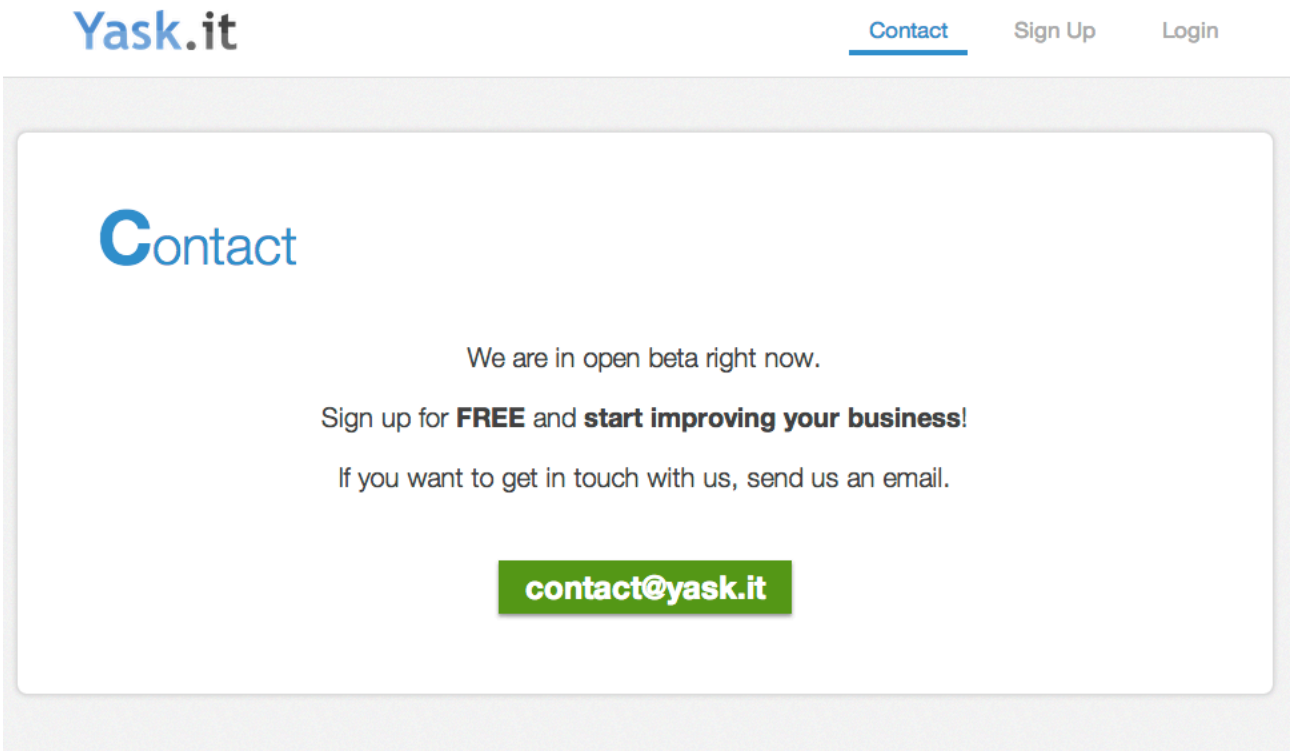

Fig 12. Vieja página de contacto

La página de contacto estaba explícitamente hecha para que la gente nos mandara un email con sus dudas o sugerencias. Sin embargo, ni un sólo correo electrónico llegó a la bandeja de entrada de *Yask.it*. Quizás nadie estaba interesado, o el diseño le alejaba. Al no tener ningún uso por parte de los visitantes, la página entera ha sido eliminada. En su lugar se han puesto dos enlaces al email de contacto por si cambia la moda y los clientes desean contactar con nosotros.

La otra página eliminada ha sido la del ejemplo de código QR para establecimientos físicos. Su objetivo era ilustrar la forma por la que sería usado el sistema. Avisaba de que era necesaria una aplicación que escanease códigos QR y sugería la mejor para cada plataforma móvil.

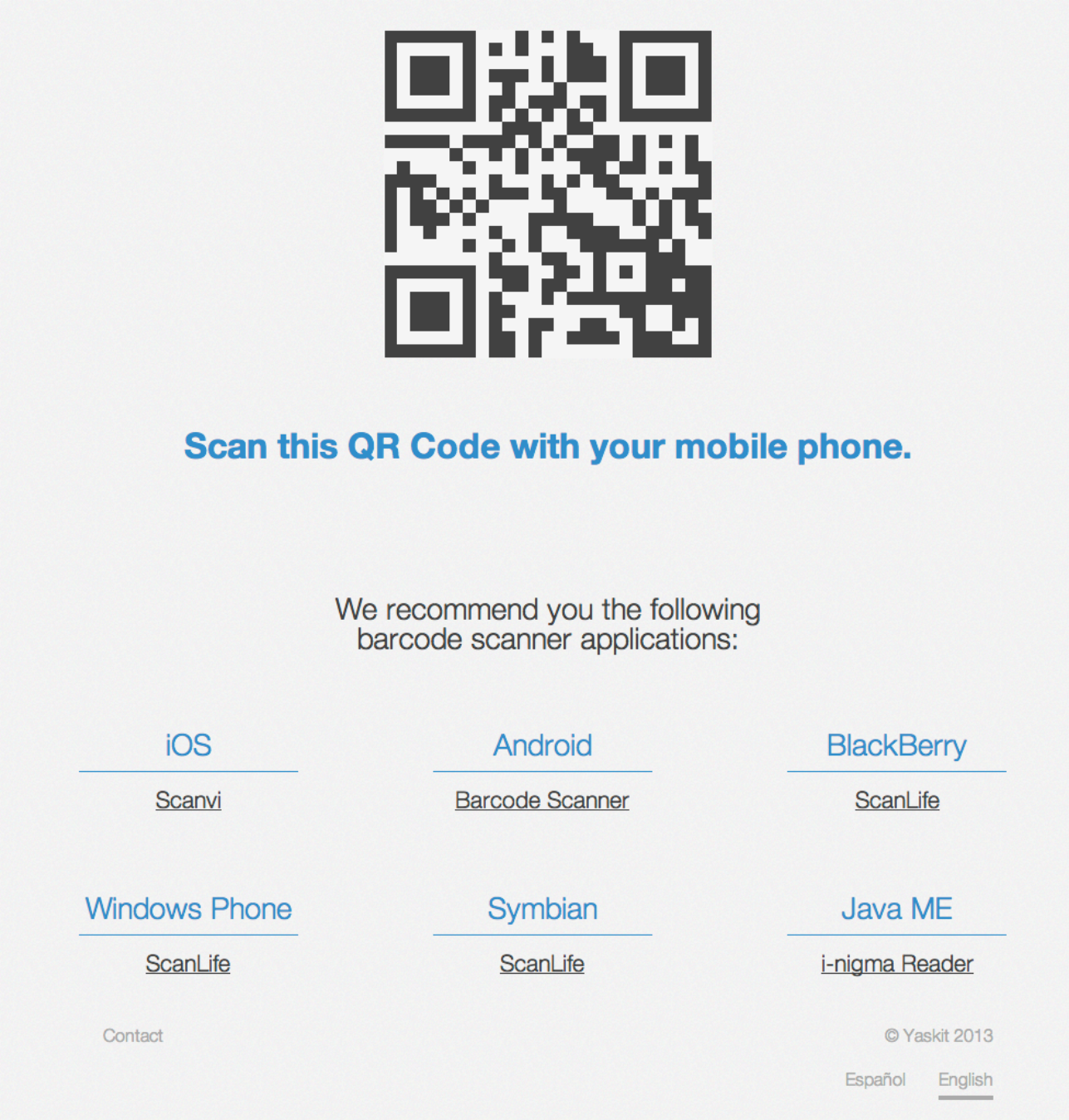

Fig 13. Vieja página de explicación de QR

Barra de navegación

La barra de navegación superior, aunque se muestra en la página principal, es una componente totalmente independiente. Durante el TFG se ha iterado múltiples veces sobre la implementación de esta barra y sus funciones.

Fig 14. Vieja barra de navegación

El objetivo de esta barra es enlazar entre las distintas secciones del sistema. En este caso entre la página principal y el panel de usuario. Además tiene enlaces para conectarte y registrarte.

En la versión anterior al hacer clic en Conectar lo que ocurría es que aparecía un diálogo modal sobre la pantalla.

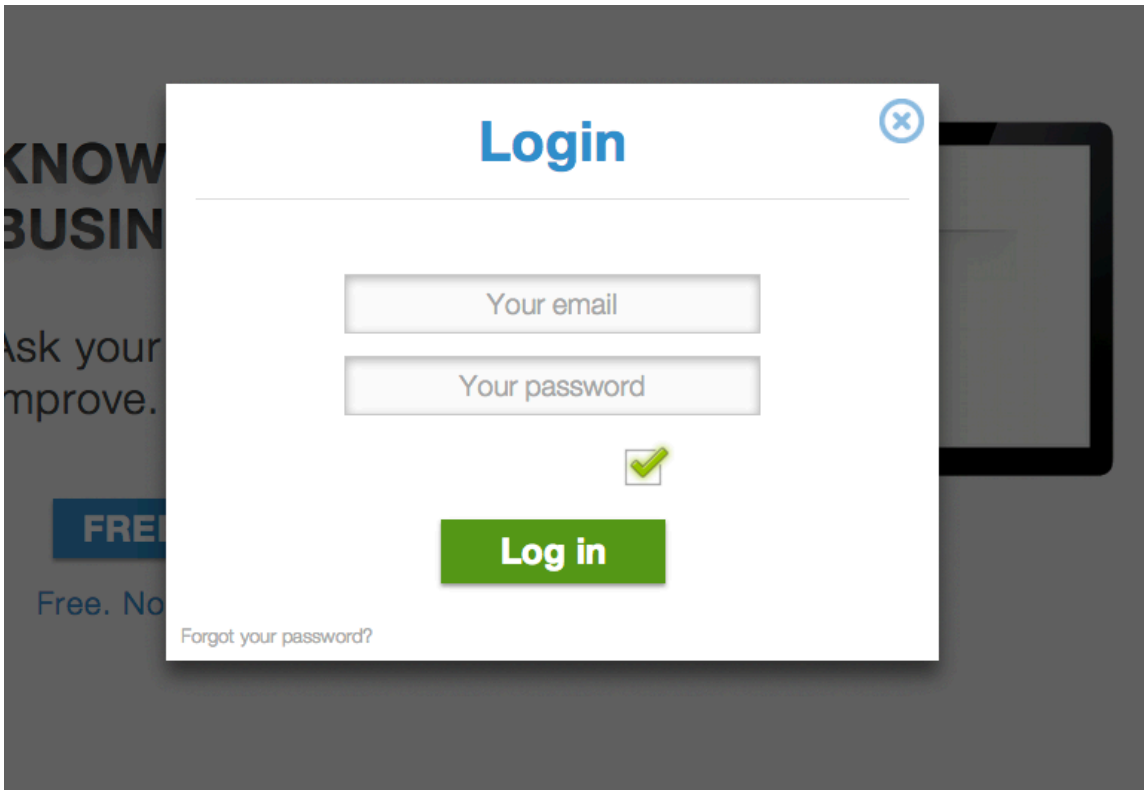

Fig 15. Viejo diálogo de Login

Este diálogo tenía ciertos problemas. En pantallas grandes funcionaba bien, pero en un dispositivo móvil no funcionaba correctamente. Al pulsar sobre un campo, el móvil suele hacer zoom sobre éste para que el usuario pueda ver que es lo que está escribiendo.

Al producirse el zoom, el diálogo se desplazaba, y el teclado en los dispositivos *Android podía llegar a* cerrarse, rompiendo totalmente la funcionalidad. Por esto este diálogo no podía quedarse así.

El soporte para móviles de la página web (aparte del sistema de encuestas) se cuida sólo en la página principal. Las demás páginas no se espera que sean visitadas desde un teléfono móvil así que no se pone tanta atención en procurar una experiencia perfecta. Pero esto no quita que se deba permitir acceder a esas secciones desde cualquier dispositivo.

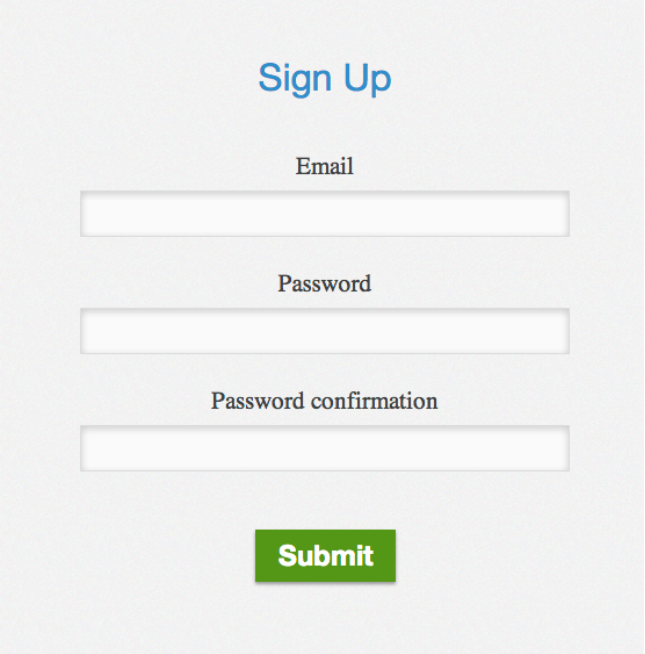

Fig 16. Vieja pantalla de registro

Por otra parte está el registro. Antes al hacer click en Registrar se abría la página de registro.

El diseño de la página de registro era soso y vacío. Le faltaba contenido. En general no motivaba a la gente para registrarse.

Por eso esta página también ha sido sustituida por algo mejor.

La idea nueva es incorporar el diálogo de *login* y la página de registro en la barra de navegación, unificando el diseño de las tres cosas en una sola.

Además, hacía falta llamar la atención hacia las cosas importantes de la barra de navegación. Las cosas importantes en este caso son el botón de Registro, y en segundo lugar, el de *Login*. Por eso los colores de la barra han sido alterados.

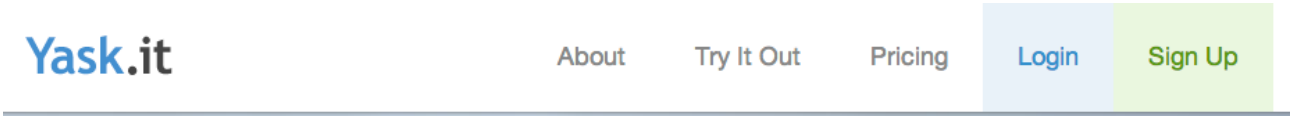

Fig 17. Nueva barra de navegación

El botón de Registro ahora es verde, con un fondo claro pero llamativo. El botón de *Login* por su parte es azul. Para remarcarse entre los otros tres botones menos importantes.

En cuanto a los diálogos de *Login* y Registro, ahora son paneles que se despliegan al hacer click.

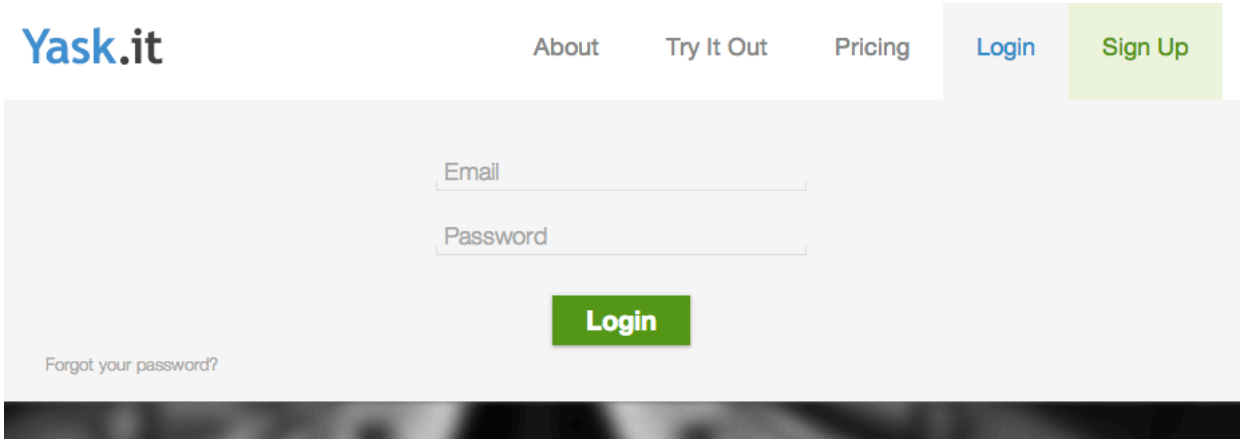

Fig 18. Nuevo panel de Login

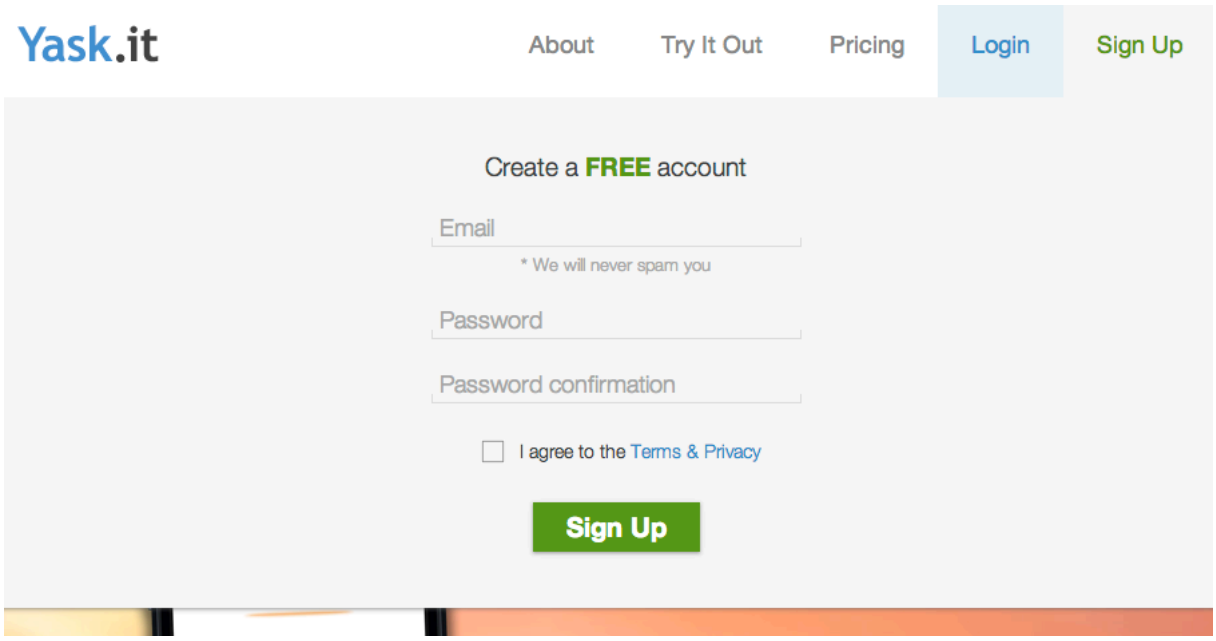

Fig 19. Nuevo panel de Registro

Estos dos paneles, al no tener tanto espacio que rellenar (una pantalla completa en el caso del viejo registro) aparentan tener más contenido. Esto hace que resulten más estéticos.

También se ha producido un cambio del diseño de los campos de texto. Se ha optado por copiar el diseño que tiene Android porque resulta elegante y minimalista.

Es interesante hablar sobre la implementación de la barra, porque se ha iterado mucho en su desarrollo. La barra de navegación se queda en la cima del documento y no se mueve de ahí, aunque los propósitos iniciales al reescribirla eran distintos.

Al principio se quería que la barra estuviera siempre en la cima de la pantalla y se mantuviera ahí por mucho que el usuario se desplazara por la página. A la hora de implementar esto en CSS, hay una solución muy sencilla:

```
#header {
    position: fixed;
    top: 0;
    left: 0;
    right: 0;
    height: 80px;
}
```

Esta solución es la más básica, ancla el elemento **#header** a la parte superior de la pantalla, hace que ocupe el 100% del ancho y mida **80px** de alto. Este método, sin embargo, trae un problema en dispositivos móviles. Al efectuarse un zoom en un dispositivo móvil, el navegador recalcula la posición y la fija de nuevo. Imposibilitando hacer zoom y moverte por el contenido de la barra de navegación (en este caso, el panel de *Login*).

La otra solución, se basa en cómo funcionan las interfaces en otros sistemas. Partir la pantalla en 2 trozos, uno para la barra de navegación y otro para el contenido. Luego a la hora de hacer scroll y desplazarse por la pantalla, efectuar todo el desplazamiento dentro del elemento con el contenido principal (como si fuera un *ScrollView*). La implementación en CSS se asemeja a:

```
#header {
    position: absolute;
    top: 0;
    left: 0;
    right: 0;
    height: 80px;
}



#content {
    position: absolute;
    top: 80px;
    left: 0;
    right: 0;
    bottom: 0;
    overflow: auto;
}
```

Al código le faltan unas cuantas reglas propietarias de *Webkit* para aumentar el rendimiento pero en esencia debería funcionar. Aunque el rendimiento final que ofrece está muy lejos de ser aceptable. Ralentiza completamente el navegador al desplazarse por la página. Además, de vez en cuando tiene problemas con la interacción en los dispositivos iPad: En vez de desplazarse por el contenedor principal, intenta desplazarse por la página entera y se queda enganchado pensando que ya ha llegado al borde de la página.

Al final se consideró esto y se optó por la opción simple de mantener la barra estática en el principio del documento. No causa problemas en dispositivos móviles como la primera opción, y ofrece un rendimiento notablemente mejor que la segunda opción.

Diseño de Correos Electrónicos

Yask.it, como la gran mayoría de servicios web, requiere una confirmación de email tras los registros. Aunque es menos estricto: permite usar el servicio durante 15 días antes de confirmar el correo. Lo que se quiere conseguir con esto es evitar que los usuarios al ver que tienen que abrir su bandeja de entrada del correo desistan y se vayan. También es posible que esto salve a usuarios que se equivocan al escribir su correo y luego desisten al ver que no les llega el correo en los próximos minutos. Pero el usuario debe confirmar la cuenta o sino, ésta se bloqueará a los 15 días.

Anteriormente estos correos tenían un diseño extremadamente básico. Se ha intentado cambiar eso (junto a la página principal y demás secciones) para mostrar una imagen más seria de proyecto.

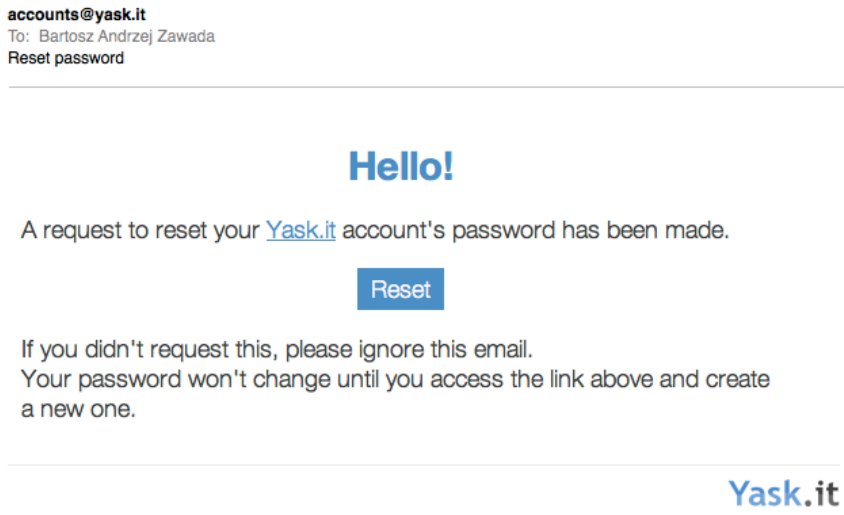

Fig 20. Viejo diseño de correos electrónicos

El nuevo diseño es más serio. Se centra en la pantalla, tiene un fondo de color y tiene algunas cosas extra como pié de página con copyright y slogan del proyecto.

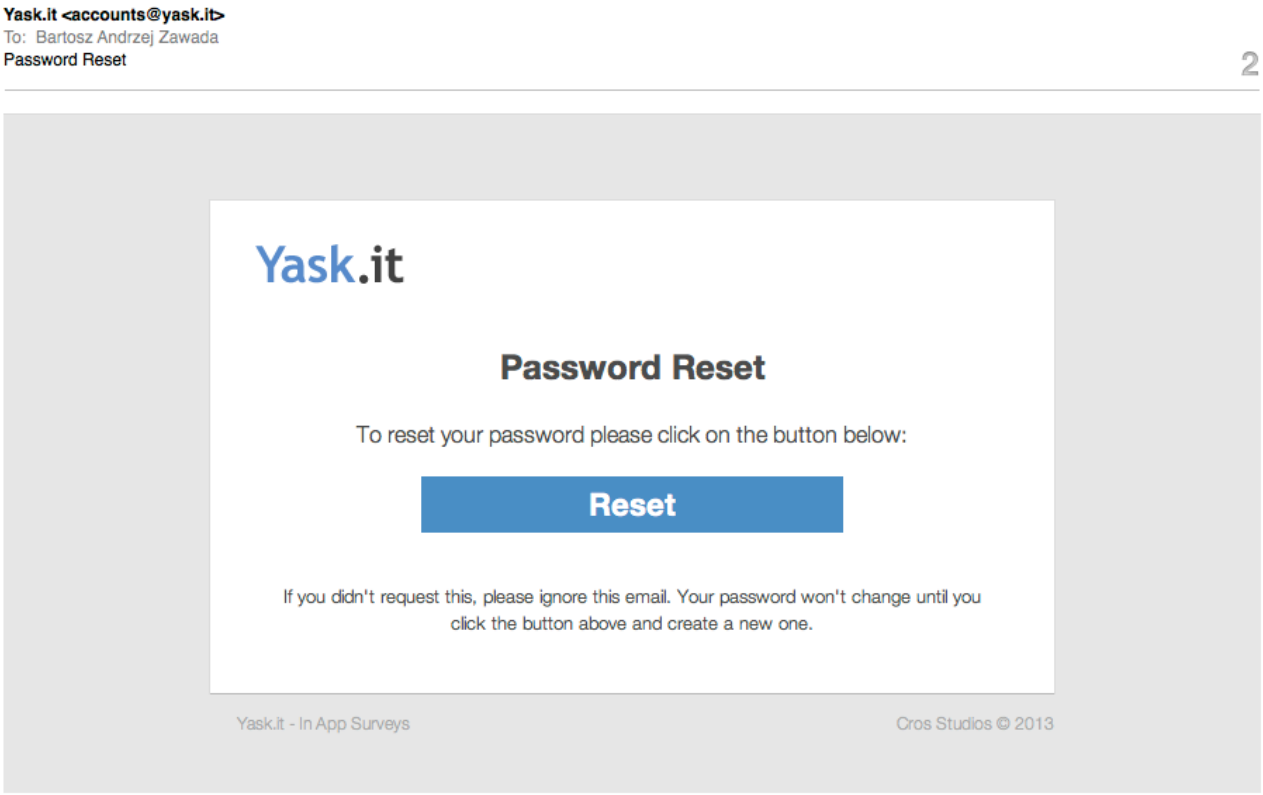

Fig 21. Nuevo diseño de correos electrónicos

La implementación de este diseño ha sido extremadamente complicada. Esto se debe a que el soporte ofrecido por los clientes de correo es terrible. Escribir emails en HTML es como escribir páginas web hace 15 años. Excepto que ni siquiera es posible poner nada en la cabecera de HTML porque los clientes web (Gmail, Yahoo! Mail) lo arrancan. La única solución es poner todos los estilos en cada elemento HTML, resultando en una tarea ardua.

Por suerte existen herramientas que simplifican el proceso. Estas herramientas permiten que escribas el CSS de forma normal, y luego ellas lo insertan *inline* en los elementos html correspondientes. Esto más o menos simplifica los estilos. Sin embargo el desarrollo sigue siendo complicado porque el marcado HTML padece de un soporte horrible obligando a escribir toda la estructura en forma de montones de tablas anidadas.

Optimización del panel de usuario

La aplicación entera realiza un uso intenso de cachés en las secciones más intensivas, como lo es el panel de usuario. Aunque el uso de cachés no te salva de tener que esperar la petición entera cuando la caché no esté cargada (o invalidada). Por eso los datos de una de las gráficas del panel de usuarios se guardaban precalculados en la base de datos. En principio esto evitaba tener que hacer un cálculo muy grande con miles de datos. Aún así, el rendimiento del panel era muy reducido. La petición completa podía llegar a tardar 2 segundos debido a los cálculos. En ese momento fue necesario realizar un *profiling* del SQL que generaba la aplicación web y cómo lo ejecutaba MySQL.

Aparentemente MySQL no es demasiado inteligente a la hora de usar índices. Teniendo índices de varias columnas definidos específicamente para esa consulta, era capaz de no usar ningún índice o usar uno subóptimo.

Tras una búsqueda por internet encontré que MySQL admite un comando `use index` que le fuerza a usar un índice específico. Sólo añadiendo esas sentencias la velocidad mejoró entre 5 y 10 veces. Haciendo que consultas que antes tardaban 120 a 200ms pasaran a tardar 15 a 30ms.

Tras ver que la velocidad volvía a ser buena, consideré deshacerme de toda la estructura de precálculo de la gráfica anterior. La estructura en sí era extremadamente compleja y causaba más problemas de los que solucionaba. Entre otras cosas hacía que el sistema fuera muy rígido.

Una de las funcionalidades que ofrece el panel de usuario es que todos los tiempos están en la zona horaria del usuario. Los recuentos de respuestas se hacen respecto a los días en la zona horaria del usuario. Las gráficas se dibujan en la misma zona horaria. El objetivo es simplificar el uso del software por parte del usuario.

Pero la estructura de precálculo de esa gráfica impedía la flexibilidad. Los datos no podían precalcularse en todas las zonas horarias. Sólo se calculaban en una.

Al final la estructura fue eliminada, simplificando el sistema. La pérdida de rendimiento resultó ser mínima. En cambio se produjo una ganancia de flexibilidad que permitió que la gráfica funcionase en cualquier zona horaria.

Más tarde aún se reconsideró la utilidad de la gráfica. La gráfica en cuestión dibujaba las medias de todas las preguntas de la encuesta en el tiempo. Con el tiempo resulta obvio que la utilidad de semejante gráfica es nula, porque las medias tienden a un número, y una vez se alcance suficiente cantidad de respuestas las medias apenas varían.

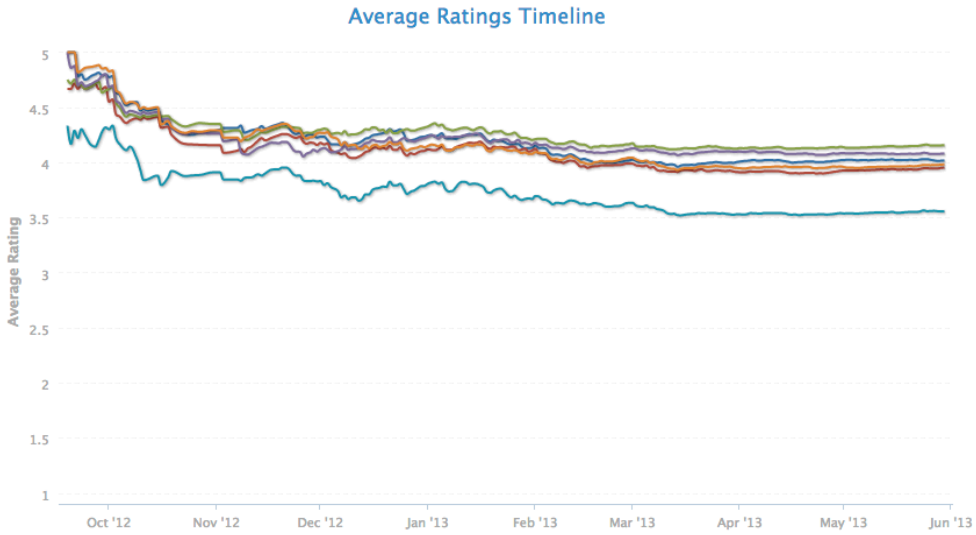

Fig 22. Vieja gráfica de medias en el tiempo

Por eso se decidió cambiar la gráfica para que mostrara algo más dinámico. Se decidió calcular las medias semanales por separado. Así puedes ver de que se quejan tus usuarios constantemente. Y cuando lances una nueva versión para resolver esas quejas ver como la puntuación media semanal ha ascendido (o disminuido).

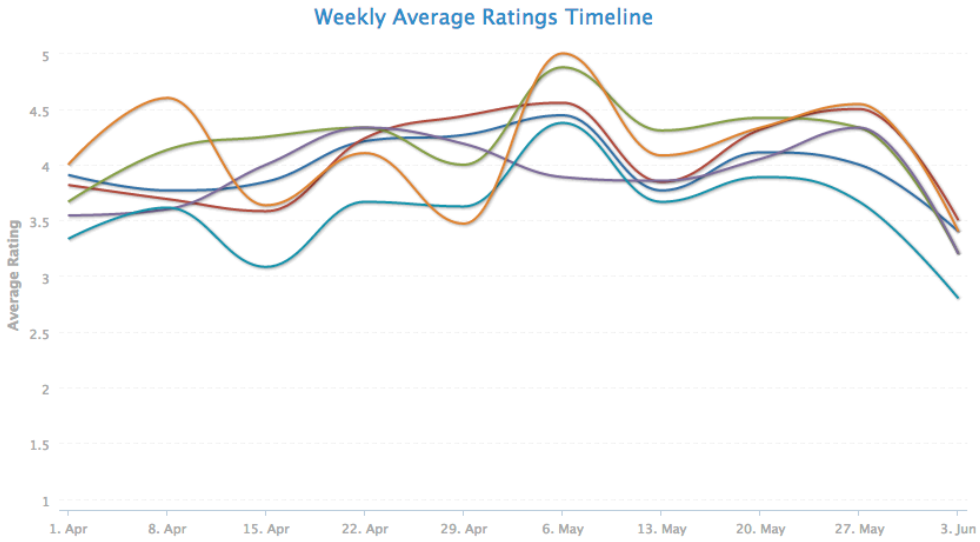

Fig 23. Nueva gráfica de medias semanales

Panel de administración

Uno de los cambios más importantes en el proyecto ha sido la creación de una panel de administración. Antes no había nada para observar la evolución del sistema. Se podía realizar consultas a la base de datos directamente, pero era lento e incómodo.

Al principio se hizo un apaño para poder observar como progresaban nuestros clientes. El apaño consistía en permitir a las cuentas de administrador elegir los sitios y encuestas de cualquier cuenta:

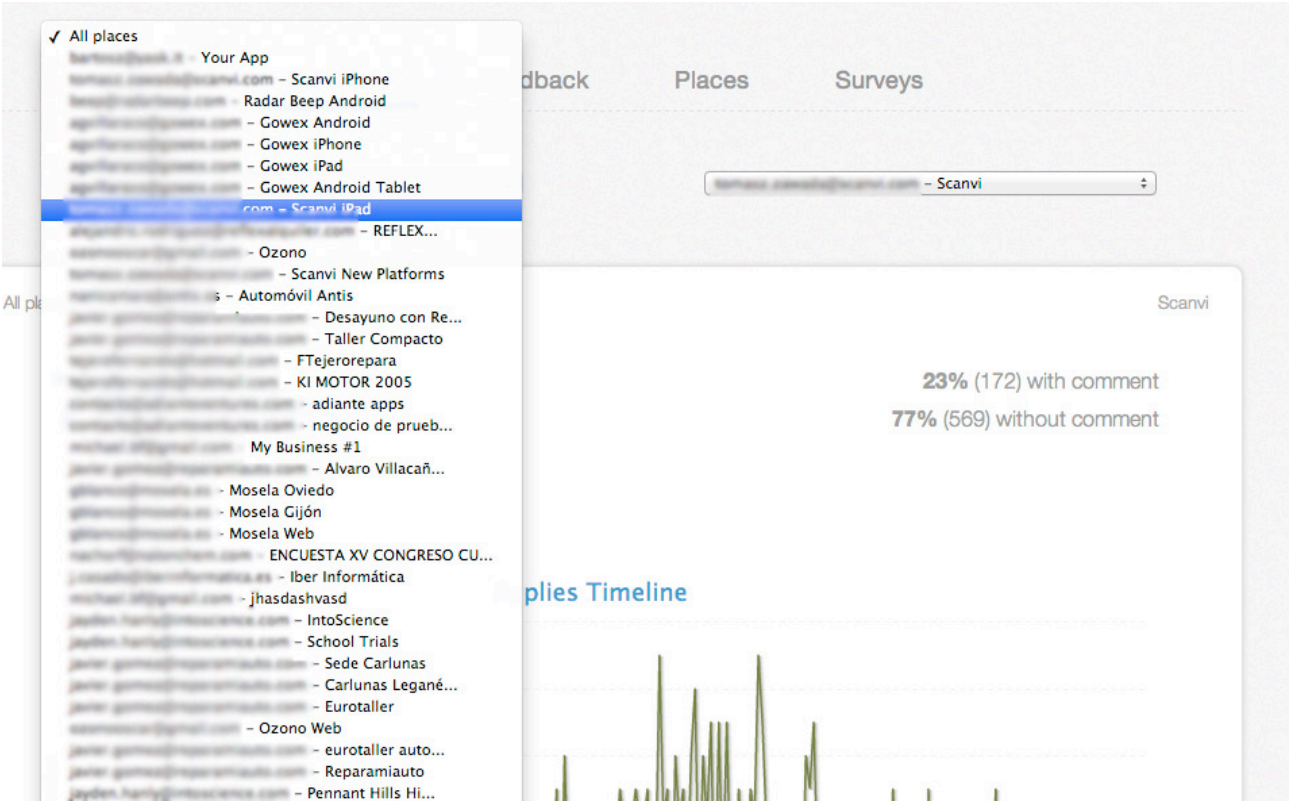

Fig 24. Primer apaño de administración

Este sistema resultaba poco práctico. No se podía saber si había ocurrido algo interesante sin pasar por cada una de las cuentas, lo que podía tomar bastante tiempo.

La solución real fue la creación de un panel específico para administración. Este panel mostraba la cantidad de respuestas globales. Además, decía la cantidad que había nuevas respuestas en los últimos 30 días, últimos 7 días y en el mismo día. Esto permite ver el flujo real de datos que mueve el sistema. Por último, ofrece la posibilidad de hacer click en cualquiera de los tres grupos anteriores para desglosarlos, mostrando quién ha recibido las respuestas y en qué sitio.

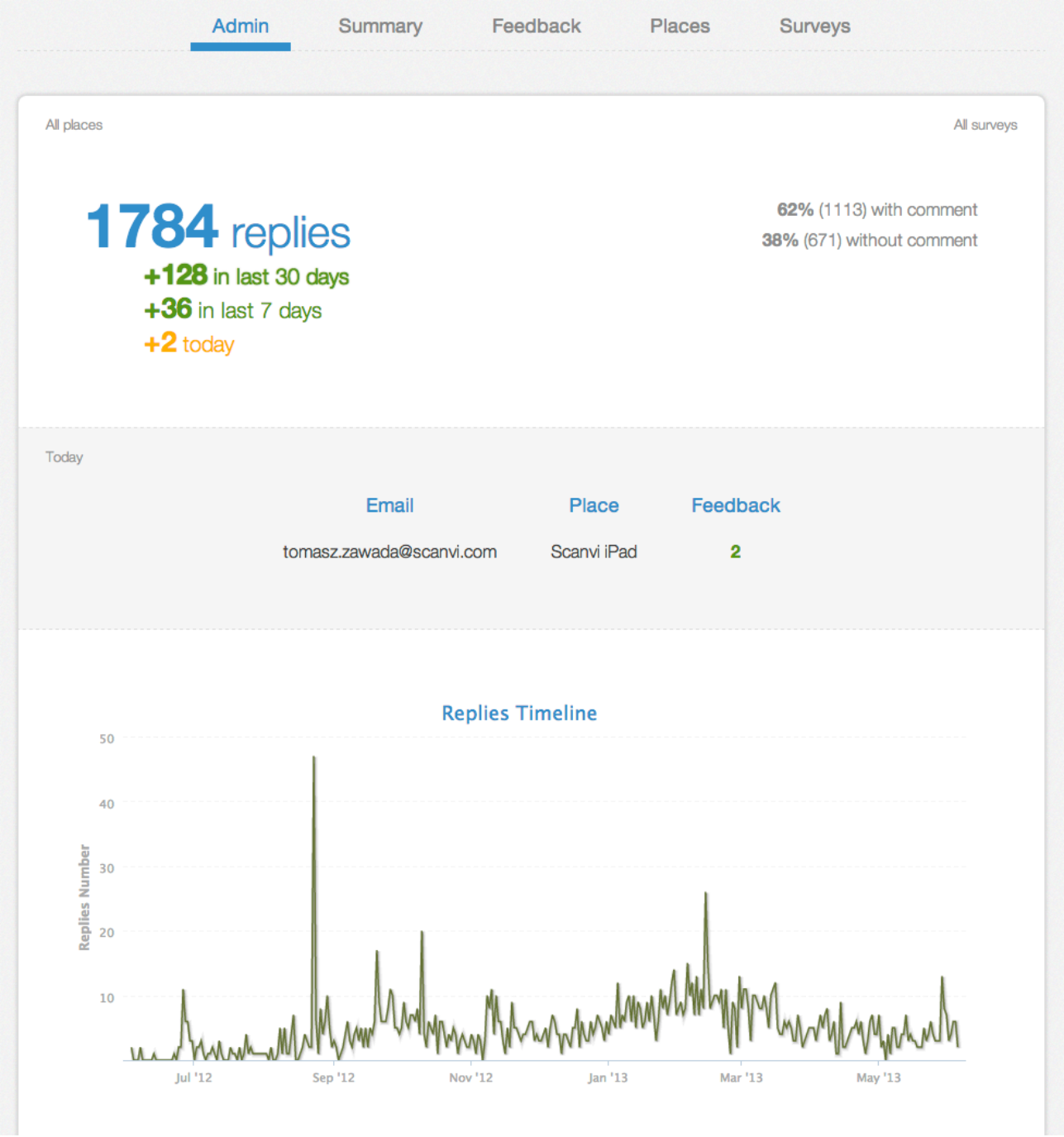

Fig 25. Parte superior del panel de administración

En la parte inferior incluye una lista de todas las cuentas con su número de respuestas, número de sitios, última fecha de *login*, y un botón que permite ver los panel de resumen y respuestas del usuario.

Página de encuestas

La página de encuestas apenas ha sufrido cambios durante la realización del TFG. El único cambio interesante ha sido la adición de pequeños indicadores textuales para las valoraciones.

Todas las preguntas no respondidas por el usuario tendrán debajo un pequeño texto en gris claro que leerá "Sin respuesta".

Cuando un usuario pulse alguna estrella, el texto cambiará para describir la valoración asignada. Además el color del texto cambiará a un azul resplandeciente y en menos de un segundo empezará a volverse transparente hasta desaparecer del todo.

El objetivo de este cambio es hacer saber a los usuarios que no marcar ninguna estrella no se corresponde con una valoración muy mala, sino con la ausencia de valoración.

Fig 26. Indicaciones en encuestas

Informando a los usuarios de esto nos aseguramos de que no hay equivocaciones entre respuestas negativas y no ausencia de respuestas.

Traducciones de comentarios

Una funcionalidad muy interesante solicitada por los clientes ha sido la posibilidad de traducir los comentarios de los usuarios porque reciben comentarios en múltiples idiomas (chino, ruso, inglés, español, etc).

La idea era utilizar alguna API de traducción gratuita. La primera idea es mirar si Google ofrece semejante API. Lo hace, pero cobra por ello y no ofrece ningún nivel gratis. Por otra parte, Microsoft tiene una API de traducción que tiene un nivel gratuito que permite traducir hasta 2 millones de caracteres al mes. Por tanto es esa la que se ha implementado.

Al principio no resulta nada fácil entender como funciona todo. Lo propio es un servicio web al que se le envien peticiones desde Javascript en el cliente pasándole una clave. Pero aparentemente eso es demasiado fácil de robar por terceros (una vez tengan la clave pueden abusar de la API a tu costa).

Para evitar eso, se usa una clave privada de la API y una ficha de acceso de corta duración. El cliente solicita al servidor web una traducción. El servidor web entonces envía una petición de ficha de acceso a los servidores de microsoft con su clave privada de API. Los servidores de Microsoft responden otorgando una ficha de acceso con una duración limitada (5 a 10 minutos generalmente) que el servidor debe mandar al cliente para que el cliente pueda solicitar traducciones a directamente a los servidores de Microsoft.

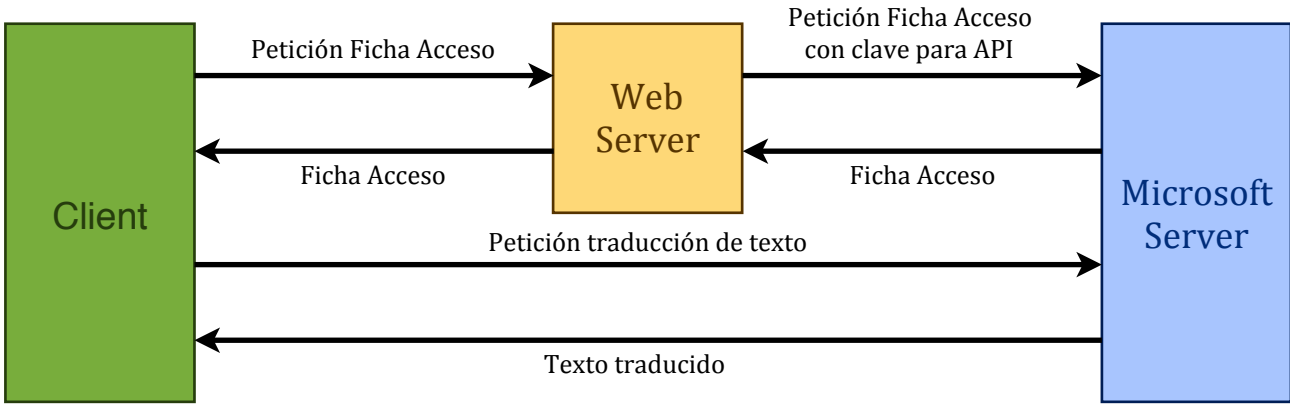

Fig 27. Diagrama de secuencia de la API de traducción de Microsoft

Migración de servidor

Se acercaba el final de la oferta de año gratuito de Amazon por lo que era el momento apropiado de considerar si seguir en Amazon o buscar otro tipo de *hosting*. En principio el EC2 de Amazon no era una buena opción para el servicio Yask.it. No hemos tenido ningún problema de tráfico por tanto poder encender instancias según la carga no era necesario. A cambio de esta posibilidad, que iba a ser desaprovechada, el coste del servicio era muy elevado, y el rendimiento pobre.

Teniendo eso en cuenta, lo mejor era buscar un server dedicado. *Hetzner*[3] ofrece muy buen equipo a buen precio, por tanto es la mejor opción. Para obtener el mismo hardware en Amazon el coste es 10 veces superior.

Durante la migración se decidió cambiar algunos componentes:

- Sustituir Apache por *Nginx* como servidor web. *Nginx* es un servidor basado en eventos, lo que significa que no se bloquea con las peticiones ofreciendo un rendimiento superior. Además consume menos memoria que Apache.

- Sustituir *MySQL* por *PostgreSQL*. *MySQL* hace varios años era mejor que *PostgreSQL*, pero esto ha cambiado. *PostgreSQL* ha mejorado bastante mientras que *MySQL* no demasiado. Además, ahora que pertenece a *Oracle* su futuro es incierto.

El cambio de Apache a *Nginx* no ha sido problemático, debido a que *Phusion Passenger* tiene soporte para ambos y hace la instalación casi automatizada. Aún así, hay que configurar *Nginx*. Por suerte, su configuración es bastante mas sencilla que la de Apache.

Por otra parte, cambiar de *MySQL* a *PostgreSQL* ha sido complicado. Lo primero el código de *Yask.it*, a pesar de ser en su mayoría agnóstico a las bases de datos, tenía ciertos trozos de código no estándar (el **`use index`** nombrado anteriormente, *SQL* no estándar para trabajar con tiempos y zonas horarias). Ha sido necesario quitar los **`use index`**, porque *PostgreSQL* no los necesita para funcionar de forma decente, y transcribir el SQL no estándar del formato de *MySQL* al formato de *PostgreSQL*. Aunque esto era la parte fácil de la migración.

La parte complicada de la migración de bases de datos es que no son compatibles de ninguna manera. Lo que sería de esperar es que si ejecutases

```
mysqldump --compatible=postgresql
```

Se crease un fichero *SQL* con el contenido de la base de datos y que *PostgreSQL* lo pudiera leer. Pero no es así. La compatibilidad entre las bases de datos es nula. Lo que se suele hacer es coger el *dump* y pasarle un script que convierta los datos al formato de la otra base de datos.

En Ruby existe una aplicación que hace exactamente eso. Se llama *mysqltopostgres*[4]. Usando esta aplicación pude pasar los datos del servidor de *Amazon* al de *Hetzner*, aunque tardé un poco de tiempo.

Mi objetivo era un cambio de servidor principal sin perder datos en ningún momento. Para ello escribí un script que realizaba la migración de datos desde el anterior servidor al nuevo en cuestión de segundos.

Lo único que podría resultar un problema es el lento cambio de las entradas del DNS. Descubrí que el DNS de *Yask.it* tenía un TTL de 3600. Así que decidí contactarles para ver si podían reducir ese tiempo. Accedieron y me ajustaron un TTL de tan sólo 60 segundos, permitiendo un cambio casi instantáneo de cualquier posible tráfico al nuevo servidor. Al día siguiente el TTL fue reiniciado a su valor original, 3600.

Términos y Privacidad

Si se compara la página de registro vieja con el nuevo panel de registro se puede observar que hay una nueva casilla que el usuario debe marcar para decir que esta de acuerdo con los términos del servicio y la política de privacidad. Lo que significa que existen:

Este apartado del sistema es accesible desde el pié de página.

Se ha decidido que redactar los términos del servicio y la política de privacidad otorgaría una imagen más seria al proyecto por lo que se han redactado ambas. Aunque sólo en inglés para disminuir la probabilidad de errores. Además son temporales; Se cambiarán cuando tenga sentido pagar a alguien que sepa de derecho para que las redacte, lo que sería en un futuro si el proyecto resulta ser comercialmente rentable.

En sí se ha partido de coger los términos de servicio y política de privacidad de otros servicios semejantes. Entre ellos *Localytics*[5] y *Emtrics*[6]. Por su parte *Localytics* tiene unos términos de servicio y política de privacidad bastante sobreprotectores, lo que es bueno. Mientras que *Emtrics* es una empresa que se dedica a lo mismo que nosotros. Por eso una mezcla de ambas ligeramente modificada puede ser una buena base.

Google Analytics

La mayor parte de sistemas grandes traen incorporados sistemas para obtener analíticas de uso. *Google Analytics* es uno de esos sistemas. Permite al administrador del sistema ver en tiempo real las características de sus usuarios y su forma de actuar: Qué botones pulsan, que secciones visitan dentro de la página web, etc. Esto es muy útil a la hora de hacer estudios para determinar si algún componente de la página cumple con su objetivo, y si se usa junto con *A/B Testing* es posible determinar como mejorar el sistema.

Por eso se ha implementado Google *Analytics* en Yask.it, para poder en el futuro entender como se comportan los visitantes de la página principal y mejorarla.

SSL

El *SSL* sirve para encriptar las comunicaciones entre cliente y servidor, aumentando la seguridad. *Yask.it* ha funcionado sin *SSL* hasta ahora. Aunque nunca está demás aumentar la seguridad. Por eso se ha decidido implementar.

En general, implementar SSL es muy sencillo en *Ruby on Rails*. Cada controlador se puede configurar por separado. En principio acepta la conexión por tanto por *HTTPS* como por *HTTP*. Pero se puede forzar a los controladores para que redirijan todo el tráfico *HTTP* a *HTTPS*.

En *Yask.it*, se han forzado casi todos los controladores excepto el que se encarga del sistema de encuestas. Porque *SSL* añade ciertas negociaciones para establecer la conexión. Estas negociaciones extra pueden ralentizar considerablemente la carga de la página en un dispositivo móvil con una conexión a internet inestable.

Conclusión

El progreso en la aplicación durante estos últimos 3 meses ha sido bastante grande y el resultado es bastante bueno. El nuevo diseño de la página principal, el correo y otros componentes le da un *look* mucho mas profesional y serio a la aplicación. Lo que seguro que repercutirá en la cantidad de gente que muestre interés.

Este proyecto no sigue un planeamiento fijo. Es un servicio que está siendo usado y evoluciona con el uso. En un principio redacté una lista de funcionalidades que tenía pensado implementar en este TFG. Sin embargo las necesidades eran distintas, los clientes deseaban funcionalidades no esperadas, así que el desarrollo se desvió del plan original. Así debe ser, porque es un proyecto vivo.

Según la evolución reciente, el siguiente paso en el desarrollo del proyecto es rehacer la página de Sitios, para mostrar el nuevo énfasis en aplicaciones móviles y, justo después, implementar la SDK nativa. En otras palabras, crear la parte del producto por la que se va a cobrar. Cuando eso esté desarrollado se verá si la aplicación tiene o no futuro en el mercado.

Referencias

[1] Panopticlick, https://panopticlick.eff.org/

[2] Evercookie, http://samy.pl/evercookie/

[3] Hetzner, http://www.hetzner.de/en/

[4] mysqltopostgres, https://github.com/ajokela/mysqltopostgres

[5] Localytics, http://www.localytics.com/

[6] Emtrics, http://emtrics.com/en/